ERROR CORRECTION DEVICE

PUB. NO.: 06-077844 [JP 6077844 A]
PUBLISHED: March 18, 1994 (19940318)
INVENTOR(s): INOUE YOSHIYUKI
NAKAMURA TAKAHIKO
KUMANO MAKOTO
ISHIMOTO JUNKO
APPLICANT(s): MITSUBISHI ELECTRIC CORP [000601] (A Japanese Company or Corporation), JP (Japan)
APPL. NO.: 05-102619 [JP 93102619]
FILED: April 28, 1993 (19930428)
INTL CLASS: [5] H03M-013/00; G11B-020/18
JAPIO CLASS: 42.4 (ELECTRONICS -- Basic Circuits); 42.5 (ELECTRONICS -- Equipment); 45.2 (INFORMATION PROCESSING -- Memory Units)
JOURNAL: Section: E, Section No. 1566, Vol. 18, No. 329, Pg. 31, June 22, 1994 (19940622)

ABSTRACT

PURPOSE: To provide an error correction device able to decode an error correction code such as a Reed Solomon code at a high speed.

CONSTITUTION: In the case of correcting an error included in reproduced data stored in a RAM 1, a syndrome generating circuit 2, a Euclid arithmetic operation circuit 20 calculating an error location polynomial and an error numeral polynomial and a chain search circuit 2 calculating an error location are operated in parallel. In the case of making divisions based on the process of Euclidean algorithm, the division on a Galois field is executed while shifting sequentially the content of a register in which coefficient data of dividend polynomial are stored.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平6－77844

(43)公開日　平成６年(1994)３月18日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｈ０３Ｍ　13/00		7259－5Ｊ		
Ｇ１１Ｂ　20/18	１０２	9074－5Ｄ		

審査請求　未請求　請求項の数９(全 52 頁)

(21)出願番号　特願平5－102619

(22)出願日　平成５年(1993)４月28日

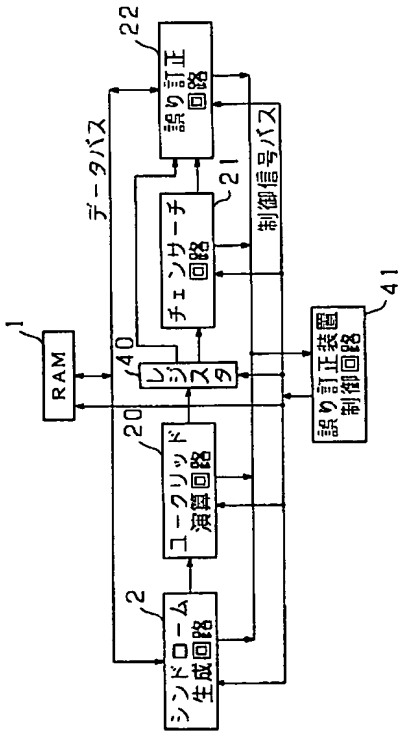
(31)優先権主張番号　特願平4－109402
(32)優先日　平４(1992)４月28日
(33)優先権主張国　日本（ＪＰ）
(31)優先権主張番号　特願平4－183742
(32)優先日　平４(1992)７月10日
(33)優先権主張国　日本（ＪＰ）

(71)出願人　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目２番３号
(72)発明者　井上　禎之
京都府長岡京市馬場図所１番地　三菱電機株式会社電子商品開発研究所内
(72)発明者　中村　隆彦
神奈川県鎌倉市大船五丁目１番１号　三菱電機株式会社情報電子研究所内
(72)発明者　熊野　真
京都府長岡京市馬場図所１番地　三菱電機株式会社電子商品開発研究所内
(74)代理人　弁理士　高田　守

最終頁に続く

(54)【発明の名称】　誤り訂正装置

(57)【要約】
【目的】　リードソロモン符号等の誤り訂正符号を高速に復号できる誤り訂正装置を提供する。
【構成】　ＲＡＭ１に蓄えられた再生データに含まれた誤りを訂正する際に、シンドローム生成回路２と、誤り位置多項式及び誤り数値多項式を算出するユークリッド演算回路20と、誤り位置を算出するチェンサーチ回路21との動作を並行して行う。ユークリッド互除過程に基づく除算を行う際に、被除多項式の係数データが記憶されているレジスタの内容を逐次シフトしながらガロア体上の除算を実行する。



1

【特許請求の範囲】

【請求項１】　入力信号に付加されている線形符号である誤り訂正符号を用いて誤り訂正を行う誤り訂正装置において、前記入力信号からシンドロームを生成するシンドローム生成手段と、該シンドローム生成手段の出力に基づき誤り位置多項式と誤り数値多項式とを演算する演算手段と、該演算手段より導出された誤り位置多項式より誤り位置を算出する誤り位置算出手段と、該誤り位置算出手段より出力される誤り位置情報と前記誤り数値多項式とを用いて誤り数値を導出して、入力信号中の誤り位置が検出された誤ったデータに誤り訂正を施す訂正手段と、前記シンドローム生成手段，前記演算手段及び前記誤り位置算出手段の３つの手段を並行して動作させる制御手段とを備えることを特徴とする誤り訂正装置。

【請求項２】　前記シンドローム生成手段及び前記誤り位置算出手段の駆動クロックの周波数を、前記演算手段を駆動するクロックの周波数より高くすることを特徴とする請求項１記載の誤り訂正装置。

【請求項３】　前記演算手段の復号ステップ数が、前記シンドローム生成手段または前記誤り位置算出手段のステップ数とほぼ等しいことを特徴とする請求項１記載の誤り訂正装置。

【請求項４】　入力信号に付加されている線形符号である誤り訂正符号を用いて誤り訂正を行う誤り訂正装置において、前記入力信号を一旦蓄える記憶手段と、前記入力信号からシンドロームを生成するシンドローム生成手段と、該シンドローム生成手段の出力に基づき誤り位置多項式と誤り数値多項式とを演算する演算手段と、該演算手段より導出された誤り位置多項式より誤り位置を算出する誤り位置算出手段と、該誤り位置算出手段より出力される誤り位置情報と前記誤り数値多項式とを用いて誤り数値を導出して、入力信号中の誤り位置が検出された誤ったデータに誤り訂正を施す訂正手段と、該訂正手段が誤り訂正を行う際、前記誤り位置情報に基づく誤った情報を少なくとも２つ以上連続して前記記憶手段より読み出した後、誤り訂正が施されたデータを前記誤り位置情報に基づく誤ったデータと逐次書き換える動作を誤り位置が検出されたデータがすべて訂正されるまで繰り返すように、前記訂正手段を制御する制御手段とを備えることを特徴とする誤り訂正装置。

【請求項５】　所定の被除多項式を除多項式で除算し、その後その剰余を除多項式とし前記除算を行なった際の除多項式を被除多項式として除算する動作を、剰余の多項式の次数がある条件を満たすまで繰り返すユークリッド互除の処理過程を含んでリード・ソロモン符号による誤り訂正処理を行なう誤り訂正装置において、前記２つの多項式の各々の係数を記憶する記憶手段と、該記憶手段に記憶されている２つの多項式の最大次数の係数同士のガロア体上の除算を行なう除算手段と、該除算手段の除数側の多項式の係数を記憶しておく前記記憶手段の出

2

力と前記除算結果とをガロア体上で乗算する乗算手段と、該乗算手段の出力と、前記除算手段の被除数側の多項式の係数を記憶しておく前記記憶手段の出力とをガロア体上で加算する加算手段とを備え、前記記憶手段をそれぞれシフトレジスタ状に配置してガロア体上の多項式同士の除算回路を構成し、被除多項式が記憶されている前記憶手段の内容を逐次シフトすることにより多項式同士の除算を実行するように構成したことを特徴とする誤り訂正装置。

【請求項６】　前記乗算手段が、ユークリッド互除法により多項式同士の除算を行なう際、初期値としてZ^{2t}がセットされる前記記憶手段側に配置されることを特徴とする請求項５記載の誤り訂正装置。

【請求項７】　前記除算手段において、除数として入力される多項式の最大次数の係数が０であった場合、前記除算手段の出力を１とすることを特徴とする請求項５または６記載の誤り訂正装置。

【請求項８】　前記除算手段の出力より誤り位置多項式を算出する際、前記除算手段に入力される被除多項式の係数が除数として扱われる場合と被除数として扱われる場合とで演算方法を切り換えることを特徴とする請求項５，６または７記載の誤り訂正装置。

【請求項９】　前記除算手段の出力と誤り位置情報とを選択する第１の選択手段と、前記２つの多項式の各々の係数を記憶しておく記憶手段の出力を選択する第２の選択手段と、前記除算手段の除数側の多項式の係数とそれより１次次数が高い前記除算手段の除数側の多項式の係数とを選択する第３の選択手段と、前記第１の選択手段の出力と前記第３の選択手段の出力とを乗算する乗算手段と、該乗算手段の出力と前記第２の選択手段の出力とを加算する加算手段とを更に備え、消失訂正時に修正シンドローム多項式を演算する際、前記第１，第２及び第３の選択手段は、それぞれ誤り位置情報，前記除算手段の除数側の多項式の係数及び１次次数が高い多項式の係数を選択するように構成したことを特徴とする請求項５，７または８記載の誤り訂正装置。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は誤り訂正符号によって符号化された情報中に発生する誤りを訂正する誤り訂正装置に関し、特に、コンパクトディスクプレーヤ（以下、ＣＤと記す）、ディジタルオーディオプレーヤ（以下、ＤＡＴと記す）、ディジタル映像信号記録再生装置（以下、ディジタルＶＴＲと記す）などで、再生または受信ディジタル信号中に発生する誤りを高速で訂正処理する誤り訂正装置に関する。

【０００２】

【従来の技術及び発明が解決しようとする課題】ＣＤ，ＤＡＴ，ディジタルＶＴＲなどに代表されるＰＣＭ（Pulse Code Modulation ）記録及び再生には誤り訂正符号

を用いた誤り訂正処理が用いられる。以下、上記ＣＤ、ＤＡＴ及びディジタルＶＴＲに実際に用いられている誤り訂正符号としてＧＦ（ｑ）上で定義されるリード・ソロモン符号を用いて誤り訂正処理を行なう誤り訂正装置について述べる。

【０００３】リード・ソロモン符号を用いて多数ワードの誤り訂正を行なうには、受信データよりシンドロームを生成して誤り位置多項式と誤り数値多項式との係数を求める必要がある。このような多項式の係数を求める一手法として、ユークリッドアルゴリズムまたはバーレカンプアルゴリズム等が知られている。例えば、上述のユークリッドアルゴリズムは与えられた２つの多項式の最大公約多項式を求めるためのアルゴリズムであって、ガロアフィールドＧＦ（ｑ）の元を係数とする多項式同士の除算を行ない、誤り位置多項式と誤り数値多項式とを求める。なお、ユークリッドアルゴリズムについての詳しい説明は後述する。

【０００４】以下、リード・ソロモン符号による多数ワードの誤り訂正の復号手順を示す。一般に、リード・ソロモン符号の復号手順は次のようになる。

（１）受信データからシンドロームS_iを求める。（シンドローム生成）

（２）求めたシンドロームＳｉから誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を求める。（ユークリッドまたはバーレカンプアルゴリズム等）

（３）誤り位置多項式σ（ｚ）より誤り位置を求める。（チェンサーチ）

（４）この誤り位置及び誤り数値多項式ω（ｚ）より誤り数値を求める。

（５）誤り位置と誤り数値とから受信データの誤りを訂正する。（誤り訂正）

【０００５】以下、ガロアフィールドＧＦ（2^8）で定義されるリード・ソロモン符号の復号方法について説明する。

【０００６】シンドロームS_iは次式で与えられる。受信多項式をＸ（ｚ）、生成多項式をＧ（ｚ）とすると、

Ｓ（ｚ）＝Ｘ（ｚ）ｍｏｄ　Ｇ（ｚ）

$＝S_0 ＋S_1 ×z＋・・・＋S_k ×z^k$

となる。なお、本従来例では符号長ｎのリード・ソロモン符号を復号する場合について説明する。

【０００７】次に、シンドロームS_i（Ｓ（ｚ））より誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を求める。以下、ユークリッドアルゴリズムを用いた誤り位置多項式及び誤り数値多項式の算出方法を簡単に説明する。ユークリッドアルゴリズムとは、ガロア体上で定義される多項式同士の除算の繰り返しであり、次のように示される。すなわち、ユークリッドアルゴリズムとは、ガロア体上で定義される２つの多項式をＡ（ｚ），Ｂ（ｚ）とすると

Ａ（ｚ）　／Ｂ（ｚ）　＝Q_0（ｚ）（商）・・・R_0（ｚ）（余り）

Ｂ（ｚ）　／R_0（ｚ）＝Q_1（ｚ）（商）・・・R_1（ｚ）（余り）

R_0（ｚ）／R_1（ｚ）＝Q_2（ｚ）（商）・・・R_2（ｚ）（余り）

というガロア体上で定義される多項式同士の除算をある一定の条件を満たすまで繰り返すものである。

【０００８】例えば、光ディスクに採用されている誤り訂正符号の場合は最小ハミング距離が17のリード・ソロモン符号が採用されているので

Ａ（ｚ）＝z^{16}

Ｂ（ｚ）＝Ｓ（ｚ）

$＝S_0 ＋S_1 ×z＋・・・＋S_{15}×z^{15}$

と表わされ、上記割り算は余りの多項式の次数が７以下になるまで繰り返される。（なお、本例では消失（イレージャ）は考慮していない。）

【０００９】図25にユークリッドアルゴリズムによる基本方程式の解（誤り位置多項式及び誤り数値多項式）を求めるフローチャートを示す。まず始めにＳ１で各値の初期値の設定を行なう。Ｓ２，Ｓ３で図示した演算を実行し、Ｓ４でＮＯの場合は各多項式をＳ５に図示したように設定し再びＳ２へ戻る。Ｓ４でＹＥＳの場合は、Ｓ６で誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を求める。なお、図中＜　＞はガウス記号を示す。

【００１０】従って、ユークリッドアルゴリズムによる誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）の算出は最小ハミング距離が17のリード・ソロモン符号の場合、ガロア体上の多項式同士の除算を最大８回行なえば良いことになる。

【００１１】ユークリッドアルゴリズムによって求められた誤り位置多項式σ（ｚ）により誤り位置を求める。これは、一般にチェンサーチという方法が用いられる。この方法は、誤り位置多項式σ（ｚ）にα（αはガロア体ＧＦ（2^8）上の元）のべき乗であるα^i（ｉ＝０，１，２，・・・，ｎ－１）を逐次代入して、σ（α^i）が０となるσ（ｚ）の根α^i（誤り位置に相当する）を求める。ただし、ｎは符号長である。

【００１２】上述の方法によると、誤り位置を演算するためには符号長に相当するＧＦ（2^8）上のｎ個の元（α^0～α^{n-1}）を逐次誤り位置多項式σ（ｚ）に代入してσ（ｚ）＝０となるα^i（誤り位置に相当する）を計算することになるので、演算のステップ数はｎ（符号長）ステップとなる。

【００１３】以下、従来の誤り訂正装置について説明する。図26は従来のリード・ソロモン符号を復号する誤り訂正装置の概略構成を示すブロック図である。図において、１は誤り訂正の対象となるデータが格納されているＲＡＭ、２は受信語よりシンドロームを生成するシンドローム生成回路、３はシンドローム生成回路２より出力

される受信シンドロームをもとに誤り訂正を行なう誤り訂正演算回路、４は誤り訂正装置全体の制御を行なう誤り訂正装置制御回路である。

【００１４】図27にはシンドローム生成回路２の具体的な回路構成図を示す。10は入力されたデータを一旦蓄えるレジスタ、11はリード・ソロモン符号の生成多項式の各係数を乗数とする乗算回路、12は加算回路である。以上レジスタ10，乗算回路11及び加算回路12でシンドローム生成回路２は構成されている。

【００１５】図28に誤り訂正演算回路３の具体的な回路構成例を示す。20は図25に示すユークリッドアルゴリズムにしたがって誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を算出するユークリッド演算回路、21はユークリッド演算回路20より出力される誤り位置多項式σ（ｚ）の係数データより誤り位置を算出するチェンサーチ回路、22はユークリッド演算回路20より出力される誤り数値多項式ω（ｚ）の係数データ及びチェンサーチ回路21より出力される誤り位置情報に基づき誤り数値を算出するとともに、誤り訂正を行なう誤り訂正回路である。以上、ユークリッド演算回路20，チェンサーチ回路21及び誤り訂正回路22で誤り訂正演算回路３は構成されている。

【００１６】図29に、ユークリッド演算回路20の上記ガロア体上の多項式の除算を行なう除算回路部分のブロック構成図を示す。50，51は入力端子、52a，52bは除数，被除数多項式の係数データを記憶しておくレジスタ、53はセレクタ、54は乗算回路、55は加算回路である。以上、レジスタ52a，52b、セレクタ53、乗算回路54及び加算回路55でユークリッド演算回路20のガロア体上の多項式の除算回路部分は構成されている。

【００１７】図30にチェンサーチ回路21の具体的な回路構成例を示す。31はチェンサーチ開始時に誤り位置多項式σ（ｚ）の係数データをセットし、その後は逐次乗算回路32の出力をセットする帰還レジスタ、32は乗算回路、33は加算回路、34は加算回路33の出力が０であるかを判定する判定回路、35は帰還レジスタ31の帰還回数をカウントする係数カウンタ、36は判定回路34より出力される０判定結果に基づき係数カウンタ35のカウント値を記憶するレジスタである。以上、帰還レジスタ31，乗算回路32，加算回路33，判定回路34，係数カウンタ35及びレジスタ36で、チェンサーチ回路21は構成されている。

【００１８】次に、図26，図27，図28，図29及び図30を用いて誤り訂正装置の動作について説明する。ＲＡＭ１に格納されている受信または再生時に発生した誤りを含む受信または再生データは、誤り訂正処理が開始されるとシンドローム生成回路２でシンドロームを生成するためにＲＡＭ１より読みだされる。シンドローム生成回路２ではＲＡＭ１より読み出されたデータ（受信多項式形式）を生成多項式で除算しシンドロームを生成する。なお、ＲＡＭ１から再生データを読み出すための読み出しアドレス、及び読み出し制御信号は誤り訂正装置制御回路４より出力される。

【００１９】図27に回路構成を示すシンドローム生成回路２はシフトレジスタ形式に配置されたレジスタ10と、入力データ（帰還データ）に生成多項式の各項に対応する係数データを乗算する乗算回路11と、加算回路12とで構成されている。シンドローム生成回路２では入力されたデータを上記演算回路で逐次演算しながらシフトして行くことによってガロア体上の多項式の除算を実行する。従って、演算を実行するに当たっては符号長分のステップ数（ｎステップ）を要する。なお、シンドローム生成回路２の詳細は、今井秀樹著「符号理論」，出版電子情報通信学会,pp.115～118 に記載されている。

【００２０】上記の要領で、シンドローム生成回路２で生成されたシンドロームは、シンドローム生成終了信号とともに、誤り訂正演算回路３及び誤り訂正装置制御回路４へ出力される。誤り訂正装置制御回路４では、シンドローム生成終了信号をもとに誤り訂正演算回路３へユークリッド演算開始信号を出力する。誤り訂正演算回路３では、ユークリッド演算開始信号が入力されると、シンドローム生成回路２より出力されたシンドロームをもとに、ユークリッド演算回路20で誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を算出する。

【００２１】以下、図29に示すユークリッド演算回路20の除算部分の回路動作を説明する。シンドローム生成終了信号が入力されると各レジスタ52には初期値がセットされる。具体的には上側のレジスタ52a 群にはZ^{2t}がセットされ、下側のレジスタ52b 群にはシンドローム生成回路２により算出されたシンドローム多項式の係数がセットされる。セットされたデータは被除数側（最初はZ^{2t}のデータがセットされた上側のレジスタ52a 群）のデータをシフトしていくことによりガロア体上の多項式の除算を実行する。除算は被除数側の多項式の次数が除数側の多項式の次数より小さくなるまで繰り返す。図29において、被除数側のレジスタ52群に接続されているセレクタ53は加算回路55の出力を選択するように接続され、除数側のレジスタ52群に接続されているセレクタ53はレジスタ52の出力を選択するように構成する。この結果被除数側のレジスタ52内のデータはシフトされ除算が実行されることになる。

【００２２】除算の結果、被除数側のレジスタ群には上記除算の余りが格納されることになる。次に除数レジスタ群に格納されている多項式を被除数として、また上記除算結果の余りを除数として除算を行なう。これは、セレクタ53の出力を切り換えることにより除数、被除数の切り換えを実行する。この動作を図25に示すアルゴリズムの終了条件が満たされるまで繰り返す。上記条件が満たされると、ユークリッド演算回路20では誤り位置多項式、及び誤り数値多項式の係数データを算出し、ユークリッド演算終了信号とともにチェンサーチ回路21、及び

誤り訂正制御回路４へ出力する。なお、ユークリッド演算回路20の除算回路部分の説明は特開平1-276825号公報に記載されている。

【００２３】誤り訂正装置制御回路４では、ユークリッド演算回路20より出力されるユークリッド演算終了信号をもとにチェンサーチ回路21の動作を開始させるチェンサーチ開始信号を出力する。チェンサーチ回路21では、チェンサーチ開始信号が入力されるとユークリッド演算回路20で算出された誤り位置多項式σ（ｚ）の各係数をもとに誤り位置を算出する。図30を用いてチェンサーチ回路21の具体的な動作を説明する。チェンサーチ回路21ではチェンサーチ開始信号が入力されると係数カウンタ35に０がセットされるとともに、帰還レジスタ31内にユークリッド演算回路20で演算された誤り位置多項式σ（ｚ）の各係数データがセットされる。そして、帰還レジスタ31の出力は乗算回路32にあらかじめセットされている乗数と乗算され、再び帰還レジスタ31でラッチされる。

【００２４】加算回路33では、各帰還レジスタ31の出力を加算する。判定回路34では、前記加算結果が０であるかを判定し、０であった場合係数カウンタ35のカウント値をレジスタ36へ書き込む。（このカウント値が誤り位置に対応する。）チェンサーチ終了にあたってはｎステップ要することになる。なお、チェンサーチ回路21の詳細は前述の「符号理論」，pp.165～166 に記載されている。

【００２５】誤り訂正回路22では、まずはじめにチェンサーチ回路21より出力される誤り位置情報をもとにＲＡＭ１内に格納されている誤ったデータを読みだす。それと同時に誤り訂正回路22内では、ユークリッド演算回路20より出力される誤り数値多項式ω（ｚ）、及びチェンサーチ回路21より出力される誤り位置情報により誤り数値を求める。ＲＡＭ１より読みだされた誤ったデータは、前述の誤り数値と加算され誤り訂正が施される。誤り訂正が施されたデータはＲＡＭ１内の誤ったデータと書き換えられる。

【００２６】上述のような誤り訂正装置を例えばディジタルＶＴＲに適用した場合の誤り訂正装置の符号、及び復号の簡単なフローを図31に示す。図31において誤り訂正装置に入力されたデータは複数個の単位情報ブロックに分割される。一般に誤り訂正符号は２重符号化されることが多く、その場合、上記複数個に分割された単位情報ブロックが複数個集められ図32に示すように２次元に配置される。そして、誤り訂正符号化装置により、垂直方向の誤り訂正符号（Ｃ２符号）が付加された後、記録方向の誤り訂正符号（Ｃ１符号）が付加される。なお、図32に示す符号構成は文献 " AN EXPERIMENTAL HOME-USE VCR WITH THREE DIMENSIONAL DCT ANDSUPERIMPOSED CORRECTION CODING" IEEE TRANSACTIONS ON CONSUMER ELECTRONICS AUG.1991 VOL.37 NUMBER3 に記載されている家庭用ディジタルＶＴＲに用いられる誤り訂正符号の１構成例である。また、図33には図32に示す記録方向の１ラインの符号構成を示す。誤り訂正符号の付加されたデータは、ディジタルＶＴＲの場合は記録媒体に記録され、通信の場合は通信路上へ送出される。記録媒体より再生されたデータは、再生過程で発生する誤りを訂正するために誤り訂正処理が施される。誤り訂正処理過程では再生データ中に含まれる誤りを訂正、または検出する。誤り訂正が施された再生データは次のステップへ出力される。

【００２７】次に、従来の誤り訂正装置で図33に示す１ラインのデータを復号する場合の各部の信号処理タイミングを図34に示すタイミングチャートを用いて説明する。図34には誤り訂正装置制御回路４より出力される各種制御信号（シンドローム生成開始信号、ユークリッド演算開始信号、チェンサーチ開始信号、及び誤り訂正開始信号）、各回路の動作タイミング（シンドローム生成回路２のシンドローム生成タイミング、ユークリッド演算回路20の演算タイミング、チェンサーチ回路21の演算タイミング、及び誤り訂正回路22の誤り訂正タイミング）、及びＲＡＭ１のデータアクセス状態を示す。誤り訂正装置制御回路４では誤り訂正が開始されるとシンドローム生成回路２へシンドローム生成開始信号を出力するとともにＲＡＭ１に格納されている再生時に発生した誤りを含む再生データを読みだすための読みだしアドレス、及び制御信号を出力する。シンドローム生成回路２ではシンドローム生成開始信号にもとづきレジスタ10内の値を０にセットするとともにＲＡＭ１より読みだされたデータに対して所定の演算（除算）処理を行う。なお、上述のようにシンドローム生成にあたっては演算終了まで符号長に相当する241 ステップを要する。この演算時間を図34中のＳＹ部に示す。

【００２８】シンドローム生成が終了すると、シンドローム生成回路２よりシンドローム生成終了信号が出力される。誤り訂正装置制御回路４ではこのシンドローム生成終了信号にもとづきユークリッド演算開始信号を出力する。ユークリッド演算回路20ではユークリッド演算開始信号にもとづき上述の要領でＧＦ（2^8）上の多項式の除算を行い、誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を算出する。なお、ユークリッドアルゴリズムを用いて、本従来例で示す最小距離17のリード・ソロモン符号を復号（誤り位置多項式、及び誤り数値多項式の算出）する場合、上述の多項式同士の除算を最大８回繰り返すことにより算出できる。この演算時間を図34中のＥＵ部に示す。この際のステップ数はユークリッド演算回路20の全体の回路構成にもよるが、図33に示す最小距離17のリード・ソロモン符号をユークリッドアルゴリズムを用いて誤り位置多項式σ（ｚ）、及び誤り数値多項式ω（ｚ）を算出するために必要なステップ数は120 ステップ程度で実現できる。回路の詳細動作の説明は

省略するが、上記除算回路での除算動作と、データのセット、制御信号の発生、誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）の算出、または本従来例では省略したがイレージャがあった場合の修正シンドロームの算出などにより約120 ステップ程度ユークリッド演算回路20での復号時間が必要になる。

【００２９】ユークリッド演算が終了するとユークリッド演算回路20よりユークリッド演算終了信号が出力される。誤り訂正装置制御回路４ではこのユークリッド終了信号にもとづきチェンサーチ開始信号を出力する。チェンサーチ回路21ではチェンサーチ開始信号に基づき係数カウンタ35に０がセットされるとともに、ユークリッド演算回路20で算出された誤り位置多項式の各係数データが各帰還レジスタ31にセットされる。そして帰還レジスタ31の出力は乗算回路32にあらかじめセットされている乗数と乗算され、再び帰還レジスタ31でラッチされる。係数カウンタ35は帰還カウンタ31の帰還数をカウントする。加算回路33では各帰還レジスタ31の出力を加算する。判定回路34では前記加算結果が０であるかを判定し、０であった場合には係数カウンタ35のカウント値をレジスタ36へ書き込む。なお、上述のようにチェンサーチ回路21では誤り位置を算出するために最大、符号長分のステップ数すなわち241 ステップを必要とする。この演算時間を図34中のＣＨ部に示す。

【００３０】チェンサーチが終了するとチェンサーチ回路21よりチェンサーチ終了信号が出力される。誤り訂正装置制御回路４ではこのチェンサーチ終了信号にもとづき誤り訂正開始信号を出力する。誤り訂正回路22の詳細なタイミングチャートを図35に示す。誤り訂正回路22では誤り訂開始信号に基づき誤り数値多項式ω（ｚ）、及びチェンサーチ回路21より出力される誤り位置情報を用いて誤り数値を算出する。一方、誤り訂正装置制御回路４では、上述の誤り位置情報にもとづきＲＡＭ１より誤ったデータを読みだす。ＲＡＭ１より読みだされた誤ったデータは、次のクロックのタイミングで誤り数値と加算され誤り訂正が施される。誤り訂正が施されたデータは次のクロックで再びＲＡＭ１へ書き込まれる。従って、誤り訂正回路22で１シンボルの誤り訂正を実行するためには３クロックの期間を最低要することになる。よって、８シンボルの誤り訂正を行うためには最低24ステップを必要とすることになる。

【００３１】誤り訂正装置は、ＣＤ，ＤＡＴまたはディジタルＶＴＲ等において再生、または受信信号中に発生する誤りを訂正するためには無くてはならない装置ではあるが従来の誤り訂正装置は以下の問題点を有する。問題点の１つは、従来の誤り訂正装置は１ラインの誤り訂正を行う際に必要とする復号時間が、図34に示すように非常に長くなってしまうという点にある。本従来例の場合、ユークリッド演算回路20の除算回路部分で高速演算のための回路アーキテクチャーを採用し高速化を図ったにも係わらず図33に示す１ラインのＣ１ブロックを復号するのに626 ステップ（241+120+241+24=626）の復号時間を要する。図29に示すような、ユークリッド演算回路20の高速化のための回路アーキテクチャはいくつか公開特許公報等で紹介されている。しかしながら、ユークリッド演算回路20での演算時間が短縮されても、シンドローム生成またはチェンサーチ等の処理時間が短縮されず、ディジタルＶＴＲのようにリアルタイムでかつ、高速信号処理を要求される場合には必要な処理速度が維持できない場合が生じる。

【００３２】誤り訂正装置の高速化に関する回路構成が特開平3-63973 号公報で紹介されている。これは、図28に示すシンドローム生成回路２及び誤り訂正演算回路３を並行動作させるものである。図36に示すタイミングチャートを用いてこの誤り訂正装置の回路動作を説明する。並行動作開始信号をもとにシンドローム生成回路２はシンドローム生成を開始する。それと同時に誤り訂正演算回路３中のユークリッド演算回路20が上述の要領でユークリッドアルゴリズムにしたがい誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を算出する。ユークリッド演算回路20での演算が終了すると誤り位置多項式の係数データをもとにチェンサーチ回路21で誤り位置の算出が行なわれる。チェンサーチ回路21で誤り位置が算出されると誤り訂正回路22で誤り訂正が施される。なお、シンドローム生成回路２、ユークリッド演算回路20、チェンサーチ回路21、及び誤り訂正回路22の動作は上記従来例と同一であるので詳細動作は説明は省略する。図36に各回路での演算ステップ数を示す。なお、図36には上記従来例と同様に誤り訂正装置制御回路４より出力される各種制御信号の出力タイミング、各回路の動作タイミング、及びＲＡＭ１のデータアクセス状態を示す。シンドローム生成回路２、及び誤り訂正演算回路３を並行動作させることにより、図33に示す１ラインのＣ１ブロックの誤り訂正を行なうために必要なステップ数は最低385 ステップ必要になる。

【００３３】しかしながら、この従来例でも図32に示すような積符号形式のリード・ソロモン符号を復号するにあたってはＣ１符号による誤り訂正終了後に、Ｃ２符号による誤り訂正を施すため復号ステップ数が非常に大きくなってしまいリアルタイムで復号できない。また、Ｃ２復号はＣ１復号が施されたデータに対して行なわれるため再度同一のＲＡＭからデータを読み出さなければならない。その結果、誤り訂正装置を並列に配置し並行処理を行うなどの対策を講じなければならず回路規模が増大してしまう等の問題点があった。

【００３４】また、図27，図30に示すように、シンドローム生成回路２及びチェンサーチ回路21はレジスタの出力にガロア体上の定数値との乗算回路と、加算回路とが直列または並列に配置されている。この際、ハードウェア上では最大で４段のＥＸ－ＯＲゲートを通ることにな

る。それに対してユークリッド演算回路20上ではハードウェア構成にもよるが、ＥＸ－ＯＲゲートを７段程度と、数個のセレクタを通るため高速化がシンドローム生成回路２及びチェンサーチ回路21に比べて困難であるという問題点があった。また、図29に示すユークリッド演算回路20の回路規模が必要以上に大きいという問題点があった。また、誤り訂正回路22での１シンボルの誤り訂正を行うためには最低３ステップを必要とし高速化において問題があった。

【００３５】最近になりディジタルＶＴＲの開発が各社で進められており、誤り訂正装置の高速化が重要な課題となってきた。図37に特開平3-172027号公報に記載されている、従来のユークリッド演算を行なう際のガロア体上で定義される多項式同士の除算回路部（以下、コア回路と記す）のブロック構成図を示す。なお、以下に説明する加算回路，乗算回路及び除算回路はガロア体上の加算，乗算及び除算演算を実行するものとして説明を続ける。

【００３６】図において、81a，81bは被除多項式，除多項式の係数データを記憶するレジスタ、82，83はレジスタ81a とレジスタ81b とのどちらかの出力を選択するセレクタ、84は除算回路、85は乗算回路、86は加算回路、95は入力端子、87，88は次数検出回路でレジスタ81a，81bに記憶されている多項式の次数を検出する。89は全体の動作の制御を行うユークリッド演算制御回路であり、セレクタ82，83の制御を行うとともに次数検出回路87，88より出力される次数判定結果によりユークリッドアルゴリズムの終了条件の判定等を行う。以上、レジスタ81a，81b、セレクタ82，83、除算回路84、乗算回路85、加算回路86、次数検出回路87，88、及びユークリッド演算制御回路89で上記ユークリッド演算回路中のガロア体上の多項式同士の除算回路部（コア回路）は構成されている。なお、本従来例では右側に配置されているレジスタに各多項式の高次の係数を記憶するように構成されている。

【００３７】次に、上記回路構成で除算を実行する際の除算回路の動作原理を簡単に説明する。いま、被除多項式及び除多項式をそれぞれ

被除多項式：$M1(Z)=A_n Z^n+A_{n-1} Z^{n-1}+\cdots+A_1 Z^1+A_0$

除多項式：$M2(Z)=B_m Z^m+B_{m-1} Z^{m-1}+\cdots+B_1 Z^1+B_0$

とする。なお、ｎ＞ｍとする。以下、筆算により上記除算を行った場合の計算手順を数１に示す。

【００３８】

【数１】

$$
\begin{array}{l}
\dfrac{A_n}{B_m}Z^{n-m}+\left(A_{n-1}+\dfrac{A_n}{B_m}B_{m-1}\right)\dfrac{1}{B_m}Z^{n-m-1}\cdots \\
B_mZ^m+B_{m-1}Z^{m-1} \overline{)\ A_nZ^n+A_{n-1}Z^{n-1}\quad+\cdots\qquad\qquad\qquad

15

$$\frac{\frac{B_m}{C_{m-1}}Z+\frac{1}{C_{m-1}}\left(B_{m-1}+\frac{B_m}{C_{m-1}}C_{m-2}\right)}{C_{m-1}Z^{m-1}+C_{m-2}Z^{m-2}+\cdots C_1Z+C_0\Big)\overline{B_mZ^m+B_{m-1}Z^{m-1}+\cdots}}\quad +B_1Z+B_0$$

$$B_mZ^m+\frac{B_m}{C_{m-1}}C_{m-2}Z^{m-1}+\cdots+\frac{B_m}{C_{m-1}}C_1Z^2+\frac{B_m}{C_{m-1}}C_0Z$$

$$\left(B_{m-1}+\frac{B_m}{C_{m-1}}C_{m-2}\right)Z^{m-1}+\cdots+\left(B_2+\frac{B_m}{C_{m-1}}C_1\right)Z^2+\left(B_1+\frac{B_m}{C_{m-1}}C_0\right)Z+B_0$$

$$\left(B_{m-1}+\frac{B_m}{C_{m-1}}C_{m-2}\right)Z^{m-1}+\cdots+\left(B_{m-1}+\frac{B_m}{C_{m-1}}C_{m-2}\right)\frac{C_2}{C_{m-1}}Z^2+\left(B_{m-1}+\frac{B_m}{C_{m-1}}C_{m-2}\right)\frac{C_1}{C_{m-1}}Z+\left(B_{m-1}+\frac{B_m}{C_{m-1}}C_{m-2}\right)\frac{C_0}{C_{m-1}}$$

$$D_{m-2}Z^{m-2}+D_{m-3}Z^{m-3}+\cdots\qquad +D_1Z+D_0$$

ただしDｉは剰余多項式の各係数とする

【００４３】なお、本筆算では便宜上、上記剰余多項式Ｒ（Ｚ）の最大次数C_{m-1}は０でないものとする。（一般に、０であった場合は図37に示す次数検出回路87，88で上記被除多項式、及び除多項式の次数を判定して被除多項式、及び除多項式の最大次数が０でなくなるまで被除多項式、及び除多項式の係数データをシフトした後、除算を実行する。また、除算演算中の剰余多項式の最大次数が０に成った場合も同様に剰余多項式の最大次数が０でなくなるまで剰余多項式の係数データをシフトした後に、除算動作を行なう。）数１と同様に被除多項式の剰余の係数が除多項式の次数よりも小さくなるまで除算動作を繰り返す。

16

【００４４】その結果の剰余多項式（Ｒ’（Ｚ）とする）の各係数データをD_i（ｉ＝０、１，２，・・・，ｍ－２）で表わす。そして、Ｒ’（Ｚ）の次数が図25に示した条件を満たしていない場合上記Ｒ（Ｚ）を被除多項式として、Ｒ’（Ｚ）を除多項式として除算を繰り返す。その際、図37の除算回路では先ほどと同様にセレクタ82，83を切り換えて除多項式と被除多項式の交換を行う。そして、この動作をユークリッドアルゴリズムの終了条件を満たすまで実行する。

【００４５】また、図38，39には従来例のガロア体上の多項式同士の除算回路動作を説明するための原理図を示す。図38，39において90は被除多項式の係数データを記憶するレジスタ、91は除多項式の係数データを記憶するレジスタ、92は除算回路、93は乗算回路、94は加算回路、95は入力端子である。以上、レジスタ90，91、除算回路92、乗算回路93、及び加算回路94でガロア体上の多項式同士の除算回路は構成されている。なお、図38，39はガロア体上の多項式同士の除算の動作を説明するための図であるため、ユークリッドアルゴリズムによる交換手段などは省略している。また、説明を簡単にするため次数検出回路は省略してある。

【００４６】以下、図38，39に示すシフトレジスタ構成でガロア体上の多項式同士の除算を行なう場合の回路動作を説明する。図38，39に示す除算回路の構成で、

被除多項式：Ａ（Ｚ）＝$Z^8+Z^7+Z^5+Z^2+1$

除多項式：Ｂ（Ｚ）＝$Z^6+Z^3+Z^2+1$

の割り算を行なう場合について説明する。

【００４７】本従来例では、被除多項式Ａ（Ｚ）の各係数がレジスタ90に、除多項式Ｂ（Ｚ）の各係数データがレジスタ91にセットされる。ガロア体上の多項式同士の除算は被除多項式Ａ（Ｚ）の各係数データが記憶されているシフトレジスタ状に配置されたレジスタ90の内容を逐次右側にシフトしながら演算する事により実行する。図38，39には上記被除多項式Ａ（Ｚ）を除多項式Ｂ（Ｚ）で除算する場合の各レジスタ内の係数データの内容を示す。

【００４８】まず始めに、図38(a)に示すようにレジスタ10に被除多項式Ａ（Ｚ）の係数がセットされ、レジスタ91に除多項式Ｂ（Ｚ）の各係数データがセットされる。なお、本従来例では右端のレジスタより高次の項の係数がセットされているものとする。図中レジスタ内に記した数字は、各レジスタにセットされた係数データを示す。図38(a)に示す状態より除算回路92，乗算回路93及び加算回路94で演算を各係数毎に行ないながら、被除多項式の係数を右へ１つだけシフトすると結果は14(b)に示すようになる。なお、除算回路92は被除多項式側の最高次数の係数を除多項式側の最高次数の係数で除算す

るように構成されている。

【００４９】各係数の状態をレジスタ内の数字で示す。すなわち、被除多項式の各係数データを記憶するレジスタ90には被除多項式を除多項式で除算を行ない被除多項式の次数を１次減らした時の剰余が自動的にセットされることになり、これが次回の除算で被除多項式となる。同様にして２回目の除算を行なうと、図39(a) に示すように剰余がレジスタ90にセットされる。同様に３回目の除算を行なった結果を図39(b) に示す。この除算動作は＊被除多項式の次数が除多項式の次数より小さくなるまで繰り返される。本従来例では被除多項式Ａ（Ｚ）と除多項式Ｂ（Ｚ）との次数差が２次であるので、上記除算動作を３回繰り返せばよい。

【００５０】以上の処理を筆算により確認する。数３に筆算でガロア体上の多項式同士の除算（上記Ａ（Ｚ）／Ｂ（Ｚ））を行なった場合の計算過程を示す。

【００５１】

【数３】

$$
\begin{array}{r}
Z^2+Z \\
\hline
Z^8+Z^3+Z^2+1 \;\big)\; Z^8+Z^7+Z^5+Z^2+1 \leftarrow (a) \\
\underline{\;Z^8+Z^5+Z^4+Z^2\;} \\
Z^7+Z^4+1 \leftarrow (b) \\
\underline{\;Z^7+Z^4+Z^3+Z\;} \\
Z^3+Z+1 \leftarrow (c),(d)
\end{array}
$$

【００５２】各段階の剰余は数３に示すように図38(a), (b),図39(a),(b) の状態に対応したものとなっている。このとき、レジスタ90に格納されているデータは被除多項式を除多項式で除算した時の剰余多項式の係数データが記憶されている。よって、図38，39に示すようなシフトレジスタ構成の除算回路を用いることによりガロア体上の多項式同士の除算を１多項式毎に行えることになる。

【００５３】以下、図25に示すユークリッドアルゴリズムに従い、リード・ソロモン符号を復号する誤り訂正装置のユークリッド演算の除算回路部の動作を図37を用いて説明する。本従来例では、光ディスク等に採用されている最小ハミング距離が17のリード・ソロモン符号を復号する場合について説明する。いま、再生信号より生成したシンドローム多項式Ｓ（Ｚ）を

$S(Z)=S_{15}Z^{15}+S_{14}Z^{14}+S_{13}Z^{13}+\cdots S_1Z+S_0$

とする。

【００５４】ユークリッド演算が開始されるとまず始め、ユークリッド演算制御回路89より出力されるユークリッド演算開始信号をもとにレジスタ81a に初期値である被除多項式 Z^{2t}（本従来例では最小距離が17の符号であるので Z^{16}）の係数が記憶されるとともに、レジスタ81b に除多項式であるシンドローム多項式Ｓ（Ｚ）の係数データが記憶される。また、次数検出回路87，88のカウンタにはそれぞれ次数の初期値である16と15とがセットされる。次数検出回路87，88内のカウンタは被除多項式の係数データを記憶しているレジスタ81a，81bが１回シフトされるたびにカウント値が１つ下がるように構成されている。

【００５５】なお、本従来例では右側に配置されているレジスタに高次の次数の係数データが記憶される。それと同時に、除多項式であるシンドローム多項式は次数検出回路88で最高次数の係数データが０であるか判定される。ユークリッド演算制御回路89では、次数検出回路88で検出された除多項式（シンドローム多項式）の最大次数の係数データの値が０の場合、除多項式の係数データが記憶されているレジスタ81b の内容をシフトし最大次数のデータが０でなくなるまでシフト動作を繰り返す。その際、次数検出回路88内の次数カウンタのカウント値が減らされていく。この動作が終了すると、セレクタ82はレジスタ81a の出力を選択するように制御され、セレクタ83はレジスタ81b の出力を選択するように制御する。

【００５６】また、被除多項式の係数データが記憶されているレジスタ81a はクロックが入力されると加算回路86の出力がラッチされるように構成されており、除多項式の係数データが記憶されているレジスタ81b は係数データが保持されるように構成されている。図37には、具体的な制御回路は省略したが、上記レジスタ81a，81bに入力されるクロック１，２を除数側の係数データが記憶されているレジスタについてのみ供給しないという方法で除多項式の係数データをレジスタ内に保持する構成を取っている。または、レジスタ81a，81bの前段にセレクタを設け、被除数側の係数の記憶されているレジスタについては加算回路86の出力を選択するように、また除数側の係数が記憶されているレジスタについては自分自身の出力をラッチするように構成することにより実現することもできる。なお、図中に記した各演算回路（除算回路84，乗算回路85及び加算回路86）はガロア体上で定義

された各演算を行うものとする。

【００５７】以下、図40，41を用いて上記従来のユークリッド演算回路中の除算回路の回路動作を説明する。図40(a) にレジスタ81a に被除多項式として初期値であるZ^{16}がセットされ、レジスタ81b に除多項式としてシンドローム多項式Ｓ（Ｚ）がセットされた状態を示す。なお、レジスタ中に記した数字及び記号はレジスタに格納されている係数データを示すものとする。本従来例では説明を簡単にするためシンドローム多項式の最高次の係数は０でないものとする。また、レジスタに記憶される各多項式の高次の次数の係数データは右端に配置されるものとする。そして、この図においては動作の説明をするため、各々のレジスタ81a，81bの出力はセレクタ82，83が選択している除算回路84，乗算回路85及び加算回路86に直接接続してある。

【００５８】図40，41に示す除算回路は図38，39で示した除算回路と同一の構成をとっており、被除多項式の剰余の次数が、除多項式の次数より小さくなるところまでシフト動作を繰り返すことにより除算を実行する。除算の終了判定は次数検出回路87，88より出力される次数情報をもとにユークリッド演算制御回路89で行う。図40(b) に上記除算が終了した際に、各レジスタ81a，81bに記憶されている係数データの状態を示す。なお、レジスタ81a に記憶されているH_{14}～H_0 は剰余多項式の係数データである。

【００５９】次に、ユークリッド演算制御回路89では、次数検出回路87（または88）より出力される上記剰余多項式の次数をもとに、剰余の次数がユークリッド演算の終了条件（本従来例では次数が７以下）を満たしていればユークリッド演算を終了し、終了条件を満たしていなければ前回の演算で除多項式であった多項式を被除多項式として、また上記剰余多項式を除多項式として演算を実行する。その際、図37に示すユークリッド演算回路中の除算部回路（コア回路）では、除多項式（剰余多項式）の最大次数が０であるかを判定し０であった場合は除多項式（剰余多項式）の最大次数が０でなくなるまでレジスタ81a の内容をシフトした後、セレクタ82，83の選択出力を切り換えてセレクタ82はレジスタ81b の出力を選択し、セレクタ83はレジスタ81a の出力を選択するように制御信号を出力する。

【００６０】図41(a) に上記の接続状態を示す。図に示すようにセレクタ82，83で各演算回路への接続を切り換えることにより被除多項式と除多項式の交換が実行されている。（なお、回路構成は図38，39に示す除算回路と同一の構成となっている。）除算は、上記例と同様に被除多項式の剰余の次数が除多項式の次数より小さくなるまで被除多項式の係数データを記憶しているレジスタの内容を演算しながらシフト動作を繰り返すことにより除算を実行する。図41(b) に上記除算が終了した際に、各レジスタ81a，81bに記憶されている係数データの状態を示す。なお、レジスタ81b に記憶されているI_{13}～I_0は剰余多項式の係数データである。

【００６１】そして、上記剰余多項式の次数を判定し所定の条件を満たしていなければセレクタ82，83の接続を切り換えてユークリッドアルゴリズムに基づく除算動作を再び行う。この除算動作を剰余多項式の次数が７次以下になるまで繰り返す。これにより、リード・ソロモン符号を復号する際に、ユークリッドアルゴリズムに基づくガロア体上の多項式同士の除算を１多項式毎に行うことができ、誤り訂正処理速度を向上させることが出来る。

【００６２】従来の誤り訂正装置のユークリッド演算回路中のガロア体上の多項式同士の除算部回路（コア回路）はユークリッド演算を高速化するため上記のように構成されているが、ユークリッド演算を行う時、多項式同士の交換を行うために各レジスタの出力がセレクタに接続されており、例えば上記従来例に示す光ディスクの誤り訂正符号として採用されている最小ハミング距離が17の符号を復号する際はセレクタが各レジスタ毎に必要となり、セレクタの制御も必要となる。このため、回路規模が必要以上に大きくなってしまい、また、これらのセレクタを制御するためセレクタ制御回路、及び制御信号用バスが必要となってくる。すなわち、特開平3-172027号公報で紹介されているユークリッド演算過程におけるガロア体上の多項式の除算部回路（コア回路）は、上記多項式の除算を高速に行うことが出来るが不必要に回路規模が大きくなり、また制御も複雑になるという問題点があった。

【００６３】本発明は斯かる事情に鑑みてなされたものであり、本発明の１つの目的は、高速動作が可能である誤り訂正装置を提供することにある。

【００６４】本発明の他の目的は、回路規模を大きくすることなく高速動作を可能とした誤り訂正装置を提供することにある。

【００６５】本発明の更に他の目的は、不必要な制御を必要としない誤り訂正装置を提供することにある。

【００６６】

【課題を解決するための手段】本願の第１発明に係る誤り訂正装置は、入力信号に付加されている線形符号である誤り訂正符号を用いて誤り訂正を行なう誤り訂正装置であって、入力信号からシンドロームを生成するためのシンドローム生成手段と、生成されたシンドロームをもとに誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を算出する誤り位置・数値多項式演算手段と、算出された誤り位置多項式σ（ｚ）を基に誤り位置を算出する誤り位置算出手段と、算出された誤り数値多項式ω（ｚ）及び誤り位置情報を基にＲＡＭ内に格納されている誤ったデータの誤り訂正を行う誤り訂正手段とを有し、誤り訂正時に、少なくともシンドローム生成手段，誤り位置・数値多項式演算手段及び誤り位置算出手段の

3つの処理動作を並行して行うように構成したものである。

【００６７】本願の第２発明に係る誤り訂正装置は、第１発明にあって、シンドローム生成手段及び誤り位置算出手段を駆動するクロックを、誤り位置・数値多項式演算手段を駆動するクロックより高い周波数のクロックとするように構成する。

【００６８】本願の第３発明に係る誤り訂正装置は、第１発明にあって、回路規模が復号時のステップ数でほぼ決定する誤り位置・数値多項式演算手段の復号ステップ数をシンドローム生成手段及び誤り位置算出手段の復号ステップ数とほぼ等しくなるように増加させるように構成する。

【００６９】本願の第４発明に係る誤り訂正装置は、第１発明にあって、誤り訂正手段によりＲＡＭ内のデータに誤り訂正を施す際に誤り位置算出手段より出力される誤り位置情報に基づく誤ったデータを少なくとも２つ以上を連続してＲＡＭより読み出した後に、誤り訂正を施したデータを逐次ＲＡＭ内の誤ったデータとに書き換え、この動作を誤り位置検出されたデータがすべて訂正されるまで繰り返すように構成する。

【００７０】本願の第５発明に係る誤り訂正装置は、所定の被除多項式を除多項式で除算し、さらにその剰余を除多項式として、また除多項式を被除多項式として除算して剰余の多項式の次数がある条件を満たすまで上記動作を繰り返すユークリッド互除の処理過程を含んでリード・ソロモン符号による誤り訂正処理を行う誤り訂正装置であって、被除多項式及び除多項式の各係数データを記憶しておくレジスタをシフトレジスタ状に配置し、そして、この２つの多項式の最大次数の係数同士を除算する除算回路と、除算回路の除数側に設定された多項式の係数データと除算回路の出力とを乗算する乗算回路と、乗算回路の出力と除算回路の被除数側に設定された多項式の係数データとを加算する加算回路とを配置し、ガロア体上の除算を実行する際に被除多項式の係数データが記憶されているレジスタの内容を逐次シフトすることによりユークリッド互除過程に基づく除算を実行するように構成する。

【００７１】本願の第６発明に係る誤り訂正装置は、第５発明において、乗算回路をユークリッド互除過程における初期値Z^{2t}をセットするレジスタ群に配置するように構成する。

【００７２】本願の第７発明に係る誤り訂正装置は、第５または第６発明において、除算回路において、除数として入力される多項式の最大次数の係数が０であった場合、除算回路の出力を１として出力するように構成する。

【００７３】本願の第８発明に係る誤り訂正装置は、第５または第６または第７発明において、除算回路より出力される除算結果を用いて誤り位置多項式σ（Ｚ）を算出する際、除算回路に入力される被除多項式の最大次数の係数データが除数として扱われる場合と被除数として扱われる場合とで算出方法を切り換えるように構成する。

【００７４】本願の第９発明に係る誤り訂正装置は、第５または第７または第８発明において、除算回路の出力と誤り位置情報とを選択する第１の選択手段と、除算回路の除数側の多項式の係数を記憶しておく記憶手段の出力と、除算回路の被除数側の多項式の係数データを記憶しておく記憶手段の出力とを選択する第２の選択手段と、除算回路の除数側の多項式の係数とそれより１次次数が高い除算回路の除数側の多項式の係数とを選択する第３の選択手段と、第１の選択手段の出力と第３の選択手段の出力とを乗算する乗算手段と、この乗算手段の出力と第２の選択手段の出力とを加算する加算手段とを有し、消失訂正時に修正シンドローム多項式を演算する際、第１，第２及び第３の選択手段でそれぞれ誤り位置情報，除算回路の除数側の多項式の係数データ及び１次次数が高い多項式の係数を選択するように構成する。

【００７５】

【作用】第１発明の誤り訂正装置は、再生または受信信号中の誤りを訂正する際、シンドローム生成手段，誤り位置・数値多項式演算手段及び誤り位置算出手段の３つの処理手段を並行して動作するように構成したので、図33に示す１誤り訂正ブロックに誤り訂正を行なう際の復号時間を半分以下にすることができる。また、復号時に高速化のネックとなる誤り訂正装置とＲＡＭとのデータのアクセスを常に行なうように構成したので、最も短い復号時間で復号できる誤り訂正装置を得ることができる。（これは、先ほども述べたように図32に示すような積符号形式の誤り訂正符号の復号を行なう場合、まずはじめに受信データをＲＡＭ内に書き込み、Ｃ１及びＣ２符号による誤り訂正を施す。このときＲＡＭ１ではシンドローム生成のためのデータの読み出し、誤り訂正を行なうための誤りが検出されたデータの読み出し及び誤り訂正が施されたデータの書き込みを行なう必要がある。誤り訂正装置の高速化を考える際、ＲＡＭと誤り訂正装置とのデータアクセスが非常に問題となっていた。）

【００７６】第２発明の誤り訂正装置は、比較的高速化が図れるシンドローム生成手段と誤り位置算出手段とのクロック周波数を誤り位置・数値多項式演算手段のクロックの周波数に比べて高くすることにより、１誤り訂正ブロックを復号するのに要する復号時間を短縮することができ処理速度を上げることが可能となる。

【００７７】第３発明の誤り訂正装置は、誤り位置・数値多項式演算手段の復号ステップ数をシンドローム生成手段または誤り位置算出手段の復号ステップ数と誤り訂正回路の復号ステップ数とほぼ等しくなるように増加させることにより、誤り位置・数値多項式演算手段の回路規模を小さくするので、誤り位置・数値多項式演算手段

の復号ステップ数は増加するものの誤り訂正装置の復号ステップ数は変化しないので誤り訂正装置の高速化を実現することができる。

【００７８】第４発明の誤り訂正装置は、誤り訂正回路の処理動作を誤り位置情報に基づきＲＡＭ内の誤ったデータを逐次読み出した後に、誤り訂正が施されたデータをＲＡＭに書き込むので、通常１シンボルの誤り訂正を行う際、ＲＡＭからのデータの読み出し、誤り訂正、ＲＡＭへのデータの書き込みと最低３ステップかかっていた動作を、ＲＡＭよりのデータの読み出し、誤り訂正及びＲＡＭへのデータの書き込みを平行動作させるので２ステップで行うことが可能となり、誤り訂正装置の高速化が図れる。特にＲＡＭとのアクセス時間が問題となる場合にＲＡＭ１とのアクセス時間が減少するので高速化が図れる。

【００７９】第５発明の誤り訂正装置は、ユークリッド互除過程におけるガロア体上の多項式同士の除算を実行する際、各多項式の係数データを交換するためのセレクタをもたずにユークリッド互除過程中の除算を実行するので、回路規模の縮小を図れるとともにセレクタ制御が不必要になり回路の簡素化が図れる。

【００８０】第６発明の誤り訂正装置は、乗算回路をユークリッド互除過程における初期値Z^{2t}をセットするレジスタ群に配置して除算動作を行うので、除算動作を行うため乗算回路の数を１つ削減して回路規模の縮小を図れる。

【００８１】第７発明の誤り訂正装置は、ユークリッド演算を行う際の除算回路中に配置されている除算回路の出力を、除数として入力される多項式の最大次数の係数が０であった場合、１とするように構成されているので、ユークリッド演算制御回路で被除多項式または除多項式の最大次数が０であった場合でも、回路規模を増大させることなくユークリッドアルゴリズムにおけるガロア体上の多項式同士の演算を実行できる。

【００８２】第８発明の誤り訂正装置は、除算回路の出力を用いて誤り位置多項式σ（Ｚ）を算出する際、除算回路に入力される被除多項式の最大次数の係数データが除数として扱われる場合と被除数として扱われる場合とで算出方法を切り換えるので、誤り位置多項式が正確に算出されるとともに、復号ステップ数を削減できる。

【００８３】第９発明の誤り訂正装置は、イレージャ訂正を行う際に算出する修正シンドローム多項式の算出をセレクタ回路を付加するだけで構成できるので、回路規模の増大を極力抑えてイレージャ訂正にも対応できる。

【００８４】

【実施例】以下、本発明をその実施例を示す図面に基づいて詳述する。

【００８５】第１実施例．図１に本発明の第１実施例である誤り訂正装置のブロック構成図を示す。図において１，２及び20～22は図26及び図28に示したものと同一であるので説明は省略する。40はユークリッド演算回路20より出力される誤り位置多項式及び誤り数値多項式の各係数データを記憶するレジスタ、41はシンドローム生成回路２，ユークリッド演算回路20，レジスタ40，チェンサーチ回路21及び誤り訂正回路22を制御するとともに、ＲＡＭ１の書き込み及び読み出し制御も行う誤り訂正装置制御回路である。なお、シンドローム生成回路２、ユークリッド演算回路20のガロア体上の多項式の除算回路部分、及びチェンサーチ回路21の回路構成は図27，図29及び図30に示すものと同一の構成であるので説明は省略する。

【００８６】次に、本発明の第１実施例である誤り訂正装置の動作を説明する。なお、本実施例では従来例と同様に図33に示す最小距離17の記録方向のＣ１符号を復号する場合について説明する。ＲＡＭ１に格納されている再生または受信時に発生した誤りを含む再生または受信データは、誤り訂正処理が開始されるとＣ１ブロック（単位情報ブロック）を単位にＲＡＭ１より読みだされる。なお、ＲＡＭ１からの再生データの読みだしは、誤り訂正装置制御回路41より出力されるシステムスタート信号に同期して発生する読みだしアドレス、及び読みだし制御信号に基づき行われる。ＲＡＭ１より読みだされたデータはシンドローム生成回路２へ出力される。シンドローム生成回路２では誤り訂正装置制御回路41より出力されるシステムスタート信号に基づきシフトレジスタ形式に配置された各レジスタ10に０をセットする。そして、ＲＡＭ１より読みだされたデータを従来例と同様に逐次演算を行いながらシフトすることにより除算動作を実行する。シンドローム生成に当たっては従来例と同様に符号長分の241 ステップを必要とする。シンドローム生成回路２で生成されたシンドロームはシンドローム生成終了信号とともにユークリッド演算回路20に出力される。シンドローム生成終了信号は誤り訂正装置制御回路41へも出力される。ユークリッド演算回路20ではシンドローム生成回路２より出力されるシンドローム生成終了信号に基づきユークリッド演算回路20での演算処理の初期値をセットする。なお、ユークリッド演算回路20の演算動作が終了していない場合にはユークリッド演算が終了と同時に演算処理の初期値をセットする。

【００８７】一方、ユークリッド演算回路20では、誤り訂正装置制御回路41より出力されるシステムスタート信号に基づきシンドローム生成終了時にあらかじめセットされた初期値を元にユークリッド演算を開始する。ユークリッド演算回路20は、従来例と同様に図25に示すアルゴリズムに従いガロア体上の多項式の除算を繰り返し行うことにより、誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）の算出を行う。図29を用いてユークリッド演算回路20の回路動作を説明する。シンドローム生成終了信号が入力されると各レジスタ52a，52bには初期値がセットされる。具体的には上側のレジスタ52a 群にはＺ

2^tがセットされ、下側のレジスタ52b 群には上記シンドローム生成回路２により算出されたシンドローム多項式の係数データがセットされる。セットされたデータはシステムスタート信号に基づき被除数側（最初はZ^{2t}のデータがセットされた上側のレジスタ52a 群）のデータをシフトしていくことによりガロア体上の多項式の除算を実行する。除算は被除数側の多項式の次数が除数側の多項式の次数より小さくなるまで繰り返す。除算動作を具体的に示すと次のようになる。図29において、被除数側のレジスタ52群に接続されているセレクタ53は加算器55の出力を選択するように接続され、除数側のレジスタ52群に接続されているセレクタ53はレジスタ52の出力を選択するように構成する。この結果、除数側のデータはレジスタ52内に維持され、被除数側のレジスタ52内のデータはシフトされ除算が実行されることになる。

【００８８】除算の結果、被除数側のレジスタ群には上記除算の余りが格納されることになる。次に除数レジスタ群に格納されている多項式を被除数として、また上記除算結果の余りを除数として除算を行なう。これは、セレクタ53の出力を切り換えることにより除数、被除数の切り換えを実行する。この動作を図25に示すユークリッドアルゴリズムの終了条件が満たされるまで繰り返す。上記、条件が満たされるとユークリッド演算回路20では誤り位置多項式及び誤り数値多項式の各係数データを算出する。算出された各係数データは一旦レジスタ40に格納される。なお、ユークリッド演算回路20は、ユークリッド演算が終了すると誤り訂正装置制御回路41へユークリッド演算終了信号を出力する。

【００８９】同様にチェンサーチ回路21では誤り訂正装置制御回路41より出力されたシステムスタート信号を基準にしてレジスタ40に記憶されている誤り位置多項式の係数データを帰還レジスタ31にセットするとともに、係数カウンタ35のカウント値を０にセットする。そして、従来例と同様に帰還レジスタ31の出力は乗算器32にあらかじめセットされている乗数と乗算され、再び帰還レジスタ31でラッチされる。また、係数カウンタ35は帰還レジスタ31の帰還回数をカウントする。

【００９０】帰還レジスタ31の出力は加算回路33で加算される。判定回路34では、前記加算結果が０であるかを判定し、０であった場合係数カウンタ35の値をレジスタ36へ書き込む。従来例と同様にこのカウント値が誤り位置に対応する。チェンサーチは符号長に相当する241 回帰還レジスタ31を動作させ誤り位置を算出する。チェンサーチが終了するとチェンサーチ終了信号が誤り訂正装置制御回路41へ出力される。誤り訂正装置制御回路41ではチェンサーチ終了信号，ユークリッド演算終了信号及びシンドローム生成終了信号にもとづき誤り訂正開始信号を誤り訂正回路22に出力する。

【００９１】誤り訂正装置制御回路41では、誤り訂正開始信号を出力すると同時に、チェンサーチ回路21より出力される誤り位置情報をもとにＲＡＭ１内に格納されている誤ったデータを逐次読みだすための読み出しアドレス及び読み出し制御信号を出力する。例えば、８誤り訂正の場合８個の誤り位置情報に相当する８個の誤った情報をＲＡＭ１より読み出す。一方、誤り訂正回路22では、入力された誤り訂正開始信号を基準にして誤り訂正を開始する。まず始めに、誤り訂正回路22内では、ユークリッド演算回路20より出力される誤り数値多項式ω（ｚ）、及びチェンサーチ回路21より出力される誤り位置情報により誤り数値を求める。ＲＡＭ１より読みだされた誤ったデータは、誤り数値と加算され誤り訂正が逐次施される。ＲＡＭ１からの誤った情報の読みだしが終了すると、上記誤り訂正が施されたデータがＲＡＭ１内の誤ったデータと逐次書き換えられる。図９に誤り訂正回路22のタイミングチャートを示す。

【００９２】次に、第１実施例を用いて誤り訂正を施す場合の、各回路部の信号処理タイミングを図２を用いて説明する。図２には、誤り訂正装置制御回路41より出力される各種制御信号の出力タイミング，各回路の動作タイミング及びＲＡＭ１のデータアクセス状態を示す。誤り訂正装置制御回路41では誤り訂正が開始されるとシステムスタート信号をシンドローム生成回路２、ユークリッド演算回路20、及びチェンサーチ回路21へ出力するとともにＲＡＭ１に格納されている再生データを読みだすための読みだしアドレス、及び制御信号を出力する。

【００９３】シンドローム生成回路２ではシステムスタート信号にもとづきレジスタ10内の値を０にセットするとともにＲＡＭ１より読みだされたデータに対して所定の演算（除算）処理を行う。なお、上述のようにシンドローム生成にあたっては演算終了まで符号長に相当する241 ステップを要する。この演算時間を図２中のＳＹ部に示す。

【００９４】シンドローム生成が終了すると、シンドローム生成回路２より生成されたシンドロームデータとともにシンドローム生成終了信号が出力される。ユークリッド演算回路20では、シンドローム生成終了信号に基づきユークリッド演算のための初期値をセットする。なお、ユークリッド演算回路20の演算が終了していない場合は、ユークリッド演算が終了しだい初期値をセットする。

【００９５】ユークリッド演算回路20ではシステムスタート信号にもとづき上述の要領でＧＦ（2^8）上の多項式の除算を行い、誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を算出する。なお、ユークリッド演算回路20で図33に示す最小距離17のリード・ソロモン符号を復号する場合、従来例と同様に約120 ステップ程度必要とする。この演算時間を図２中のＥＵ部に示す。

【００９６】ユークリッド演算回路20によって算出された誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）の各係数データはレジスタ40に一旦記憶される。なお、

ユークリッド演算が終了するとユークリッド演算回路20よりユークリッド演算終了信号が出力される。

【００９７】チェンサーチ回路21ではシステムスタート信号に基づき係数カウンタ35に０がセットされるとともに、ユークリッド演算回路20で算出された誤り位置多項式の各係数データが各帰還レジスタ31にセットされる。そして帰還レジスタ31の出力は乗算回路32にあらかじめセットされている乗数と乗算され、再び帰還レジスタ31でラッチされる。係数カウンタ35は帰還カウンタ31の帰還数をカウントする。加算回路33では各帰還レジスタ31の出力を加算する。判定回路34では前記加算結果が０であるかを判定し、０であった場合には係数カウンタ35のカウント値をレジスタ36へ書き込む。なお、上述のようにチェンサーチ回路21では誤り位置を算出するために最大、符号長分のステップ数すなわち241 ステップを必要とする。この演算時間を図２中のＣＨ部に示す。

【００９８】チェンサーチが終了するとチェンサーチ回路21よりチェンサーチ終了信号が出力される。誤り訂正装置制御回路41ではチェンサーチ終了信号，ユークリッド演算終了信号及びシンドローム生成終了信号にもとづき誤り訂正開始信号を出力する。誤り訂正回路22では誤り訂正開始信号に基づき誤り数値多項式ω（ｚ）、及びチェンサーチ回路21より出力される誤り位置情報を用いて誤り数値を算出する。そして、ＲＡＭ１より読み出された誤り位置を算出されたデータに誤り数値を加算することにより誤り訂正を施す。誤り訂正が施されたデータは、ＲＡＭ１からの誤ったデータの読み出しが終了すると同時にＲＡＭ１へ書き込まれる。図９に誤り訂正回路22のタイミングチャートを示す。本実施例では８シンボルの誤りを訂正するために16ステップを要する。なお、図９に示す誤り訂正回路のタイミングチャートについての詳しい説明は後述する。

【００９９】誤り訂正装置制御回路41では、ＲＡＭ１，シンドローム生成回路２，ユークリッド演算回路20，レジスタ40，チェンサーチ回路21及び誤り訂正回路22の上記動作を制御する。その際、シンドローム生成回路２，ユークリッド演算回路20及びチェンサーチ回路21の３つの回路の並行動作を管理するため、誤り訂正装置制御回路41では、システムスタート信号で各々の回路の動作フラグをセットし、各々の回路より出力される演算終了信号に基づき上記動作フラグをリセットする動作フラグ生成回路を内部に持っており、このフラグの状態により各回路を制御する。具体的には、この動作フラグの状態により、ＲＡＭ１のアドレス制御、各々の回路で算出されたデータの転送、または各々の回路の初期値のセット等の制御を行うための信号を各回路へ出力する。また、この動作フラグは各回路へも出力される。

【０１００】誤り訂正開始信号はシンドローム生成回路２とＲＡＭ１とのアクセス、及びチェンサーチ回路21での誤り位置の計算が終了すると同時に出力される。それと同時にチェンサーチ回路21で算出された誤り位置情報を基に、ＲＡＭ１より誤ったデータを読み出すための読みだしアドレス及び読みだし制御信号を発生する。

【０１０１】本実施例に示すように誤り訂正装置において、シンドローム生成回路２，ユークリッド演算回路20及びチェンサーチ回路21の３つの回路を並行動作することにより、図２に示すように１誤り訂正ブロックを復号するステップ数を257 ステップとすることができ、従来の誤り訂正装置に比べて約2.4 倍（626 /257）に復号速度を上げることが可能である。また、シンドローム生成回路２と誤り訂正演算回路３とを並行に動作せる場合に比べても1.5 倍（386 /257）に復号速度を上げることが可能である。よって、誤り訂正装置を並列に配置することなく回路の高速化をはかることができ、並列に誤り訂正装置を配置した場合に比べて回路規模の大幅な縮小を図れる。

【０１０２】また、本実施例のように並行処理を行う誤り訂正装置においては、ＲＡＭ１のデータアクセス時間が誤り訂正装置の動作速度を決定する。特に図32に示す積符号形式の誤り訂正符号を復号する際は、再生された誤りを含むデータを一旦ＲＡＭ１内にデータを格納した後に誤り訂正を施す。一般に、積符号形式の誤り訂正符号の復号はＣ１符号により記録方向の誤り訂正を行なった後にＣ２符号により垂直方向の誤り訂正を行ない、Ｃ１符号により誤りを訂正できなかったデータに対してＣ２符号により誤り訂正を行なう。（記録方向に付加されたＣ１符号による誤り訂正を行なう場合は、ＲＡＭ１内にデータを一旦格納せずに、再生データより直接シンドロームを生成して誤り訂正を施してもよいが、Ｃ２符号による誤り訂正を行なう際は図32に示す誤り訂正ブロックのデータを垂直方向に読み出す必要がある。よって、Ｃ２符号による再生データの誤り訂正はＲＡＭ１内に一旦蓄えられたデータに対して誤り訂正が施される。）

【０１０３】ＲＡＭ１内に一旦格納された再生データは、誤り訂正符号による誤り訂正を行なうためＲＡＭ１より読み出される。ＲＡＭ１より読み出されたデータは上述のように誤り訂正装置でシンドローム生成、誤り位置の算出、及び誤り数値の算出が行なわれる。そして、算出された誤り位置に相当するアドレスのデータがＲＡＭ１より読み出され、誤り数値と加算され誤り訂正が施される。誤り訂正が施されたデータはＲＡＭ１内の誤り位置に相当するアドレスへ書き込まれる。

【０１０４】この際使用されるＲＡＭ１は高速にデータをランダムにアクセスできるＳＲＡＭ、またはＤＲＡＭなどが用いられる。一般に前述のようなＲＡＭはデータの入出力Ｉ／Ｏポート、及び書き込み及び読み出しのアドレスポートは別々には設けられていない。従って、誤り訂正装置において最高速度の処理を実現するためにはＲＡＭ１とのデータのアクセスがあるシンドローム生成回路２及び誤り訂正回路22をそれぞれ独立に動作させ、

ＲＡＭ１とのアクセスがないユークリッド演算回路20及びチェンサーチ回路21をシンドローム生成回路２と並行動作させればよい。すなわち、シンドローム生成回路２及び誤り訂正回路22が動作している間に、ユークリッド演算回路20及びチェンサーチ回路21の演算を終了するように誤り訂正装置を動作させればよいことになる。このように誤り訂正装置を動作させればＲＡＭ１は、誤り訂正動作が開始されると常にシンドローム生成回路２または誤り訂正回路22とアクセス状態にあり、最小の復号時間で誤り訂正を施すことができる。なお、本実施例におけるＲＡＭ１の動作状態を示すＲＡＭアクセス信号を図２に示す。従って、第１実施例の構成をとる誤り訂正装置（すなわち、シンドローム生成回路２，ユークリッド演算回路20及びチェンサーチ回路21の３つの回路を並行動作させる構成）が最高速の復号速度を有することになる。

【０１０５】第２実施例．次に、本発明の第２実施例について説明する。図３は第２実施例である誤り訂正装置のブロック構成図である。図において１，２，20～22，40，41は第１実施例に示したものと同一のものであるので説明は省略する。本実施例では、図に示すようにシンドローム生成回路２，チェンサーチ回路21及び誤り訂正回路22はクロック１で駆動するように構成されている。また、ユークリッド演算回路20は上記クロック１と異なる周波数のクロック２で駆動するように構成されている。なお、誤り訂正装置制御回路41はクロック１及びクロック２をもとにＲＡＭ１，シンドローム生成回路２，ユークリッド演算回路20，レジスタ40，チェンサーチ回路21及び誤り訂正回路22を制御する。また、シンドローム生成回路２、ユークリッド演算回路20のガロア体上の多項式の除算回路、及びチェンサーチ回路21の回路構成は図27，図29及び図30に示すものと同一の構成であるので説明は省略する。

【０１０６】次に、本発明の第２実施例である誤り訂正装置の動作を説明する。本実施例ではクロック１の周波数をクロック２の周波数の２倍の周波数で制御した場合について説明する。これは、図27及び図30に示すように、シンドローム生成回路２及びチェンサーチ回路21はレジスタの出力にガロア体上の定数値との乗算回路と、加算回路が直列または並列に配置されている。この際、ハードウェア上ではレジスタ間で最大で４段のＥＸ－ＯＲゲートを通ることになる。それに対してユークリッド演算回路20上ではハードウェア構成にもよるが、変数どうしの乗算回路と加算器で構成されているのでＥＸ－ＯＲゲートを７段程度と、数個のセレクタを通って次のレジスタにデータがラッチされる。従って、このシンドローム生成回路２及びチェンサーチ回路21はユークリッド演算回路20に比べて高速化が比較的容易である。なお、説明をわかりやすくするためクロック２の周波数は上記実施例１で回路を駆動したクロックと同一の周波数であるとする。また、本実施例では第１実施例と同様に図33に示す最小距離17の記録方向のＣ１符号を復号する場合について説明する。

【０１０７】ＲＡＭ１に格納されている再生、または受信データは誤り訂正処理が開始されるとＣ１ブロック（単位情報ブロック）を単位にＲＡＭ１より読みだされる。なお、ＲＡＭ１からのデータの読みだしは、誤り訂正装置制御回路41より出力されるシステムスタート信号に基準にしてシンドローム生成回路２を駆動するクロック１に同期して発生する読みだしアドレス及び読みだし制御信号に基づき行われる。ＲＡＭ１より読みだされたデータはシンドローム生成回路２へ出力される。

【０１０８】シンドローム生成回路２では、第１実施例と同様に、シンドロームが生成されるが、この時の動作は第１実施例と同じであるので説明は省略する。シンドローム生成に当たっては第１実施例と同様に符号長分の241 ステップを必要とするが、シンドローム生成回路２を駆動するクロックは第１実施例に比べて２倍の周波数のクロックで駆動しているので、第１実施例の半分の時間でシンドロームの生成は終了する。ユークリッド演算回路20では、誤り訂正装置制御回路41より出力されるシステムスタート信号に基づきシンドローム演算終了時にあらかじめセットされた初期値を元にクロック２に同期してユークリッド演算を開始する。ユークリッド演算回路20の動作は、第１実施例と同様であり、その説明は省略する。その後、第１実施例と同様に、チェンサーチ回路21において誤り位置が算出される。なお、チェンサーチ回路21を駆動するクロックは上述のように、第１実施例に比べて２倍の周波数のクロックで駆動しているので、第１実施例の半分の時間で誤り位置の算出が終了する。その後、第１実施例と同様に、誤り訂正回路22で誤り数値が求められ、誤り訂正が逐次施され、誤り訂正が施されたデータがＲＡＭ１内の誤ったデータと逐次書き換えられる。

【０１０９】次に、第２実施例を用いて図33に示す１ラインのデータを復号する場合の各回路部の信号処理タイミングを図４に示す。図４において、図２と同一部分には同一符号を付している。

【０１１０】シンドローム生成にあたっては演算終了まで符号長に相当する241 ステップを要するが、シンドローム生成にあたってはクロック１の周波数が第１実施例の場合と比べて２倍になっているのでシンドローム生成にかかる時間（図４中のＳＹ部）は半分となる（241/２ステップ程度）。なお、ユークリッド演算部分のクロック周波数は第１実施例と同じであるので、その演算時間（図４中のＥＵ部）は第１実施例と同じである（120 ステップ程度）。一方、チェンサーチ回路21を駆動するクロックの周波数はシンドローム生成の場合と同様に第１実施例の場合の２倍であるので、チェンサーチ回路21での演算時間（図４中のＣＨ部）は第１実施例に比べて半

分の時間となる（241/２ステップ程度）。また、第２実施例では誤り訂正回路22をクロック１で駆動しているので、処理時間は第１実施例に比べて半分の処理時間で動作は完了する（８ステップ程度）。

【０１１１】第２実施例では、シンドローム生成回路２，ユークリッド演算回路20及びチェンサーチ回路21の３つの回路の並行動作を、クロック１及びクロック２の２系統で制御することとし、かつシンドローム生成回路２及びチェンサーチ回路21を駆動するクロックの周波数をユークリッド演算回路20を駆動するクロックの周波数に対して２倍の周波数に設定する。従って、図４に示すように１誤り訂正ブロックを復号するステップ数を128.5 ステップとすることができ、従来の誤り訂正装置に比べて約4.8 倍（626 /128.5）に上げることが可能である。また、シンドローム生成回路２と誤り訂正演算回路３とを並行に動作せる場合に比べても約３倍（386 /128.5）に復号速度を上げることが可能である。よって、誤り訂正装置を並列に配置することなく回路の高速化をはかることができ、並列に誤り訂正装置を配置した場合に比べて回路規模の大幅な縮小を図れる。

【０１１２】また、第１実施例と同様、本実施例でも、ＲＡＭ１とのデータのアクセスがないユークリッド演算回路20及びチェンサーチ回路21を、シンドローム生成回路２と並行動作させ、シンドローム生成回路２及び誤り訂正回路22が動作している間に、ユークリッド演算回路20及びチェンサーチ回路21の演算が終了するように動作させているので、最小の復号時間で誤り訂正を施すことができる。更に、本実施例の誤り訂正装置では、比較的高速化が可能なシンドローム生成回路２及びチェンサーチ回路21を駆動するクロックを第１実施例に比べて２倍の周波数のクロックにしたため、第１実施例に比べておよそ半分の復号時間で１ブロック当たりの誤り訂正動作を行うことが可能となる。

【０１１３】なお、第２実施例では、クロック１の周波数をクロック２の周波数の２倍の周波数に設定したが、これに限るものではなく例えば３倍，４倍等の整数倍、または1.5 倍，3.2 倍等の非整数倍に選んでも同様の効果を奏する。また、シンドローム生成回路２及びチェンサーチ回路21を同一周波数のクロックで駆動したがこれに限るものではなく、ユークリッド演算回路20を駆動するクロックの周波数より高いクロックで駆動すれば、シンドローム生成回路２を駆動するクロックとチェンサーチ回路21を駆動するクロックの周波数が異なってもよい。更に、誤り訂正回路22を駆動するクロックとしてクロック１を用いたがこれに限るものではなくクロック２または他の周波数のクロックを用いてもよい。

【０１１４】第３実施例．次に、第３実施例について説明する。なお、本発明の第３実施例である誤り訂正装置のブロック構成は第１実施例（図１）と同様である。図５に第３実施例におけるユークリッド演算回路20のガロア体上の多項式の除算を実行する除算回路部分のブロック構成図を示す。

【０１１５】図５において、50及び51は図29に示すものと同一であるので説明は省略する。60，65及び66はレジスタ、61及び62はセレクタ、64は加算器、63は乗算回路である。以上、レジスタ60，65及び66、セレクタ61及び62、加算器64、及び乗算回路63でユークリッド演算回路20は構成されている。なお、本実施例では、図29に示す従来のユークリッド演算回路20の除算回路に比べて図５に示すユークリッド演算回路20の除算回路は各レジスタ毎に接続されていた乗算回路がレジスタ２個毎に１個接続されており、回路規模の削減を図っている。具体的にはセレクタ62及びレジスタ66は増えているが、例えば本実施例のように最小距離17のリード・ソロモン符号を復号する場合、乗算回路が16個(400×16＝6400ゲート）が省略でき、増加するレジスタ66は８個（８×56＝448 ゲート）、及びセレクタ62が16個（24×16＝384 ゲート）となり約5500ゲートの回路規模の縮小が図れる。

【０１１６】次に、第３実施例である誤り訂正装置の動作を説明する。本実施例では第１実施例と同様に図33に示す最小距離17の記録方向のＣ１符号を復号する場合について説明する。なお、第３実施例におけるシンドローム生成回路２，チェンサーチ回路21及び誤り訂正回路22の動作は第１実施例と同様であるのでその説明は省略し、本実施例の特徴部分であるユークリッド演算回路20の動作についてのみ図５を用いて詳述する。

【０１１７】シンドローム生成終了信号が入力されると各レジスタ60，65には初期値がセットされる。具体的には図５に示す上側２列のレジスタ群60a，60bにはZ^{2t}がセットされ、下側２列のレジスタ群65a，65bにはシンドローム生成回路２により算出されたシンドローム多項式の係数がセットされる。セットされたデータはシステムスタート信号に基づき被除数側（最初はZ^{2t}のデータがセットされた上側２列のレジスタ群60）のデータをシフトしていくことによりガロア体上の多項式の除算を実行する。除算は被除数側の多項式の次数が除数側の多項式の次数より小さくなるまで繰り返す。

【０１１８】除算動作を具体的に示すと次のようになる。図５において、被除数側のレジスタ60群に接続されているセレクタ61はレジスタ66の出力を選択するように接続され、除数側のレジスタ65群に接続されているセレクタ61はレジスタ65の出力を選択するように構成する。この結果、除数側のデータはレジスタ65内に維持され、被除数側のレジスタ60内のデータはシフトされ除算が実行されることになる。

【０１１９】この際、本実施例に示すユークリッド演算回路20の除算回路は図に示すように乗算回路63がレジスタ２個に１つ配置されているためセレクタ62を逐次切り換えていくことにより除算を実行する。具体的には、被除数側の一方のレジスタのデータが除算を行なっている

間、他方のレジスタはデータを保持するようにセレクタ61を制御する。図６に被除数側の各レジスタ60，除数側の各レジスタ65，レジスタ66及び乗算回路63の動作を示すタイミングチャートを示す。なお、本タイミングチャートはレジスタ60に被除数が、レジスタ65に除数が記憶されている場合のものである。なお、除数、被除数が入れ替わっても制御のタイミング（演算順序、またはステップ数）は変わらない。各レジスタ、及びセレクタの制御は図６に示すタイミングチャートにしたがって実行される。図に示すように、ガロア体上の除算を実行するに当たっては４ステップで被除数の次数を２次減らすように構成されている。（従来例では２ステップで被除数の次数を２次減らすように構成されている。）従って、本実施例ではユークリッド演算回路20での復号ステップ数は約240 ステップ程度になる。

【０１２０】除算の結果、被除数側のレジスタ群には上記除算の余りが格納されることになる。次に除数レジスタ群に格納されている多項式を被除数として、また上記除算結果の余りを除数として除算を行なう。これは、セレクタ群61の制御を切り換えることにより除数、被除数の切り換えを実行する。この動作を図25に示すユークリッドアルゴリズムの終了条件が満たされるまで繰り返す。上記、条件が満たされるとユークリッド演算回路20では誤り位置多項式、及び誤り数値多項式の各係数データを算出する。算出された各係数データは一旦レジスタ40に格納される。なお、ユークリッド演算回路20は、ユークリッド演算が終了すると誤り訂正装置制御回路41へユークリッド演算終了信号を出力する。

【０１２１】次に、第３実施例を用いて図33に示す１ラインのデータを復号する場合の各回路部の信号処理タイミングを図７に示す。図７において、図２と同一部分には同一符号を付している。本実施例では、ユークリッドアルゴリズムを用いて、図33に示す最小距離17のリード・ソロモン符号を復号する場合、ユークリッド演算回路20における演算時間（図７中のＥＵ部）に約240 ステップ程度必要とする。他の部分は第１実施例と同一である。

【０１２２】本実施例では、第１実施例と同様に、従来例に比べて復号速度を上げることができる。また、ユークリッド演算回路20の復号ステップ数をシンドローム生成回路２及び誤り訂正回路22を動作させる際に必要なステップ数程度に増加させる（この際、シンドローム生成、及び誤り訂正に必要となるステップ数とほぼ等しくなるようにステップ数を決定する）ことにより、誤り訂正装置の復号時間を遅くすることなく回路規模の削減が図れる。本実施例では従来例に比べて約5500ゲートの回路規模の縮小が図れる。よって、誤り訂正装置を並列に配置することなく回路の高速化を図ることができ、並列に誤り訂正装置を配置した場合に比べて回路規模の大幅な縮小が図れる。

【０１２３】なお、本実施例ではユークリッド演算回路20の回路規模を、最大復号ステップ数がほぼチェンーサーチ、またはシンドローム生成とほぼ等しくなるように決定したが、これに限るものではなく、復号時間に余裕があればその分を更なる回路規模の削減に用いても良いことは言うまでもない。また、ＲＡＭ１とのデータのアクセスがないユークリッド演算回路20及びチェンサーチ回路21を、シンドローム生成回路２と並行動作させることにより、最小の復号時間で誤り訂正を施すことができる点は、第１，第２実施例と同様である。

【０１２４】第４実施例．次に、第４実施例について説明する。なお、第４実施例は上記第３実施例である誤り訂正装置中のユークリッド演算回路20のガロア体上の多項式の除算を実行する除算回路の変形例に関するものである。図８に第４実施例によるユークリッド演算回路20の除算回路部分のブロック構成図を示す。なお、本実施例の基本構成は図37に示すものである。

【０１２５】図８において、50及び51は図29に示すものと同一であるので説明は省略する。70及び71はレジスタ、72，73及び74はセレクタ、75は乗算回路、76は加算器、77はレジスタ、78は除算回路である。以上、レジスタ70，71、セレクタ72，73，74、乗算回路75、加算器76、レジスタ77、及び除算回路78でユークリッド演算回路20のガロア体上の多項式の除算回路は構成されている。なお、本実施例では、被除数の最大次数の係数を、除数の最大次数の係数で除算した結果を各除数の係数に乗算することにより割り算を実行する。図８に示すユークリッド演算回路20の除算回路はセレクタ74で被除数、及び除数を選択した後、除数側の各係数データに上記除算回路78の出力を乗算し被除数側の各係数データを加算することによりガロア体上の多項式の除算を実行する。

【０１２６】よって、第４実施例では、被除数側に接続されていた乗算回路をすべて削除するとともに、除数側のレジスタに接続されている乗算回路をレジスタ２個毎に１個接続することにより第３実施例と同様に回路規模の削減を図っている。具体的には第３実施例に比べて、セレクタ74及び除算回路78が増えているが、最小距離17のリード・ソロモン符号を復号する場合、乗算回路が７個(400×７＝2800ゲート）が省略でき、増加するセレクタ74は16個（24×16＝384 ゲート）、及び除算回路78が１個（2000ゲート程度）となり約400 ゲート程度の回路規模の縮小が図れる。

【０１２７】次に、本発明の第４実施例である誤り訂正装置の動作を説明する。本実施例では第１実施例と同様に図33に示す最小距離17の記録方向のＣ１符号を復号する場合について説明する。なお、第４実施例におけるシンドローム生成回路２，チェンサーチ回路21及び誤り訂正回路22の動作は第１実施例と同様であるのでその説明は省略し、本実施例の特徴部分であるユークリッド演算回路20の動作についてのみ図８を用いて詳述する。

【０１２８】シンドローム生成終了信号が入力されると各レジスタ70，71には初期値がセットされる。具体的には図８に示す上側２列のレジスタ群70a，70bにはZ^{2t}がセットされ、下側２列のレジスタ群71a，71bにはシンドローム生成回路２により算出されたシンドローム多項式の係数がセットされる。セットされたデータはシステムスタート信号に基づき被除数側（最初はZ^{2t}のデータがセットされた上側２列のレジスタ群70）のデータをシフトしていくことによりガロア体上の多項式の除算を実行する。除算は被除数側の多項式の次数が除数側の多項式の次数より小さくなるまで繰り返す。

【０１２９】除算動作を具体的に示すと次のようになる。図８において、被除数側のレジスタ70群に接続されているセレクタ72a はレジスタ77の出力を選択するように接続され、除数側のレジスタ71群に接続されているセレクタ72b はレジスタ71の出力を選択するように構成する。この結果、除数側のデータはレジスタ71内に維持され、被除数側のレジスタ70内のデータは逐次シフトされ除算が実行されることになる。

【０１３０】この際、本実施例に示すユークリッド演算回路20の除算回路は、除数の各係数に除算回路78の出力データを乗算するが、図に示すように乗算回路75はレジスタ２個に１つ配置されているためセレクタ73を逐次切り換えていくことにより除算を実行する。具体的には、被除数側の一方のレジスタのデータが除算を行なっている間、他方のレジスタはデータを保持するようにセレクタ72を制御する。回路全体の制御は第３実施例と同様になされる（図６参照）。なお、本タイミングチャートはレジスタ70に被除数が、レジスタ71に除数が記憶されている場合のものである。

【０１３１】なお、第４実施例では誤り訂正装置の復号時間を遅くすることなく回路規模の削減が図れるなど、本実施例における他の動作，効果などは、前述の第３実施例と同じであるので、その説明は省略する。

【０１３２】なお、第３実施例（第４実施例）ではレジスタ60，65（70，71）内のデータを保持するためにレジスタ60，65（70，71）の全段にセレクタ61（72）を設けたがこれに限るものではなく、例えばデータを保持するレジスタ60，65（70，71）に入力されるクロックを止めるなどの処理を施しても同様の効果を奏する。また、入力端子50の出力に接続されている１段目の乗算回路63（75）及びレジスタ66（77）は入力端子50より入力されるデータが０であるので省略することができることはいうまでもない。また、ユークリッド演算回路20のガロア体上の多項式の復号ステップ数を２倍にした場合の回路構成について述べたがこれに限るものではなく、シンドローム生成回路２及び誤り訂正回路22を動作させる際に必要なステップ数とほぼ等しくなるような復号ステップ数であれば３倍または４倍等のステップ数で復号できる回路を用いてもよい。更に、ユークリッド演算回路20の復号ステップ数をシンドローム生成回路２、及び誤り訂正回路22を動作させる際に必要なステップ数とほぼ等しくなるように設定したが、復号時間に余裕があればこれに限るものではなくもっと復号ステップ数を増やしても同様の効果を奏する。

【０１３３】次に、上述の各実施例に共通である誤り訂正回路22の回路動作について、図９及び図10を用いて説明する。誤り訂正回路22では、上述のようにチェンサーチ回路21より出力される誤り位置情報と、ユークリッド演算回路20で算出された誤り数値多項式とをもとにＲＡＭ１内の誤り位置が算出されたデータに誤り訂正を施す。図９に示すタイミングチャートをもとに誤り訂正回路22の回路動作を説明する。チェンサーチ回路21より出力された誤り位置情報に基づき誤り訂正装置制御回路41はＲＡＭ１内の誤り位置の算出されたデータを読み出すための読みだしアドレス及び制御信号を出力する。この際、ＲＡＭ１内の誤ったデータは図９に示すように逐次８個連続して読み出される。

【０１３４】ＲＡＭ１より連続して読み出された誤ったデータは、誤り訂正回路22で計算された誤り数値と図９に示す誤り訂正タイミングで逐次加算される。誤り訂正が施されたデータは、ＲＡＭ１からの誤り位置に算出されたすべてのデータの読み出しが終了すると同時に誤ったデータが記憶されているアドレスへ逐次書き込まれる。８誤り訂正の場合、誤り訂正回路22の回路動作のステップ数は16となり従来最低24ステップ必要であったのに対し８クロック分高速化が可能となる。

【０１３５】同様に図10のタイミングチャートを用いて誤り訂正回路22の他の動作を説明する。この例では、チェンサーチ回路21より出力された誤り位置情報に基づき誤り訂正装置制御回路41はＲＡＭ１内の誤り位置の算出されたデータを読み出すための読みだしアドレス、及び制御信号を出力する。この際、ＲＡＭ１内の誤ったデータは図10に示すように２つ連続して読み出される。

【０１３６】ＲＡＭ１より２つ連続して読み出された誤ったデータは、誤り訂正回路22で計算された誤り数値と図10に示す誤り訂正タイミングで逐次加算される。誤り訂正が施されたデータは、図10に示すタイミングで２つ連続して誤り位置が算出されたデータが記憶されているアドレスへ書き込まれる。以下、この動作を誤りを訂正するデータの数だけ繰り返して誤り訂正を行う。図10に示すような８誤り訂正の場合、誤り訂正回路22の回路動作のステップ数は16となり従来最低24ステップ必要であったのに対し８クロック分高速化が可能となる。

【０１３７】一般に誤り訂正装置の高速化を実現する際、上記実施例でも述べたようにＲＡＭ１とのデータのアクセス時間がいちばんネックとなる。これは、他の部分は上記実施例に示すように各々の回路動作を並行化することにより復号時間の短縮が図れるが、ＲＡＭ１とのデータのアクセスは並行処理が出来ないため、高速化を

実現する際ＲＡＭ１とのアクセスがあるシンドローム生成回路２及び誤り訂正回路22の計算ステップ数を如何に減らすかがキーワードとなってくる。本実施例に示す誤り訂正回路22を採用した場合、図33に示す最小距離17のＣ１符号を誤り訂正する最低際に必要なステップ数は(241＋16）ステップ程度となる。なお、本実施例では８シンボルの誤りを訂正する際の動作について説明したがこれに限るものではない。また、ＲＡＭ１より誤ったデータを、すべてまたは２つ連続して読み出したがこれに限るものではなく、ＲＡＭ１より誤ったデータ３つを連続して読み出した後に誤り訂正が施されたデータをＲＡＭ１へ書き込む等、連続して読み出すデータの数が２以上であれば同様の効果を奏することはいうまでもない。

【０１３８】なお、上記第１～第４実施例では、シンドローム生成回路２で求められたシンドロームより誤り位置多項式及び誤り数値多項式を求める方法としてユークリッドアルゴリズムを用いた場合について説明したがこれに限るものではなく、バーレカンプアルゴリズムのような他の方法により誤り位置多項式、及び誤り数値多項式を求めても、シンドローム生成回路２，ユークリッド演算回路20及びチェンサーチ回路21を並行動作させることにより高速に誤り訂正符号を復号できることはいうまでもない。

【０１３９】また、誤り訂正符号としてリード・ソロモン符号を用いたがこれに限るものではなく、ＢＣＨ符号などの他の線形誤り訂正符号を復号する誤り訂正装置に用いてもよい。

【０１４０】また、図33に示すＣ１符号の復号について説明したがこれに限るものではなくＣ２符号の復号にも同様の回路構成で復号すれば同様の効果を奏することはいうまでもない。誤り訂正符号の構成は図32及び図33に示すものに限るものではなく、また２重以上の積符号形式の誤り訂正符号を復号する場合も同様の効果を奏する。また、積符号形式の誤り訂正符号としてリード・ソロモン符号とＢＣＨ符号とを組み合わせた符号などでも上記構成で復号回路の高速化が図れる。

【０１４１】また、シンドローム生成回路２，ユークリッド演算回路20及びチェンサーチ回路21の３つの回路をシステムスタート信号に基づき同時に動作させるように制御したが、並行動作を行うタイミングはこれに限るものではなく、例えばシンドローム生成後すぐにユークリッド演算回路20の初期値をセットしユークリッド演算回路20を動作させてもよい。また、チェンサーチ回路21をユークリッド演算終了後すぐに動作させてもよい。シンドローム生成回路２より出力されるシンドロームデータの出力タイミング、ユークリッド演算回路20での初期値セットタイミング、レジスタ41への誤り位置多項式及び誤り数値多項式の係数データの出力タイミング、チェンサーチ回路21の初期値セットタイミング、誤り訂正回路22での制御タイミングは前述したタイミングに限るものではなく、シンドローム生成回路２，ユークリッド演算回路20及びチェンサーチ回路21の３つの回路を並行動作させることにより同様の効果を奏する。

【０１４２】また、シンドローム生成回路２，ユークリッド演算回路20のガロア体上の除算回路及びチェンサーチ回路21の回路構成を図27，図29及び図30に示したがこれに限るものではなく、例えば演算回路部分をマイコン等の演算処理回路等で、または大容量のＲＯＭ等で構成してもよい。

【０１４３】また、シンドローム生成回路２，ユークリッド演算回路20及びチェンサーチ回路21の３つの回路を並行動作させたがこれに限るものではなく、例えばこれらの３つの回路以外に誤り訂正回路22を含むて４つの回路を並行動作させても同様の効果を奏する。

【０１４４】また、誤り訂正回路22の回路動作はこれに限るものではなく、例えば、復号時間に余裕があれば従来例に示すようにチェンサーチ回路21で検出された誤り位置に基づく誤ったデータをＲＡＭ１より読みだし誤り訂正を行い、再びＲＡＭ１へ書き込むといった動作を繰り返してもよい。

【０１４５】また、ユークリッド演算回路20の出力を一旦レジスタ40に蓄えたが誤り位置多項式、誤り数値多項式の係数を蓄えるレジスタは必ずしも必要がなく、制御方法によっては省略してもよい。

【０１４６】また、シンドローム生成回路２，ユークリッド演算回路20及びチェンサーチ回路21の回路動作の開始を同時に行っていたがこれに限るものではなく、開始タイミングは上記３つの回路がバラバラであっても同様の効果を奏する。

【０１４７】次に、ガロア体上で定義される多項式同士の除算回路部の構成に特徴がある本発明の他の実施例（第５～第９実施例）について説明する。

【０１４８】第５実施例．図11に本発明の第５実施例である誤り訂正装置のガロア体上で定義される多項式同士の除算回路部のブロック構成図を示す。図において81a，81b，85～88及び95は図37に示すものと同一であるので説明は省略する。100 はレジスタ81a の出力を被除数として、またレジスタ81b の出力を除数として除算を行う除算回路、101 はユークリッド演算の除算回路部の制御を行うユークリッド演算制御回路である。

【０１４９】次に、図11に示す回路構成で除算を実行する際の除算回路の動作原理を簡単に説明する。いま、被除多項式、及び除多項式を従来例と同様にそれぞれ

被除多項式：$M1(Z)=A_n Z^n+A_{n-1} Z^{n-1}+\cdots A_1 Z+A_0$

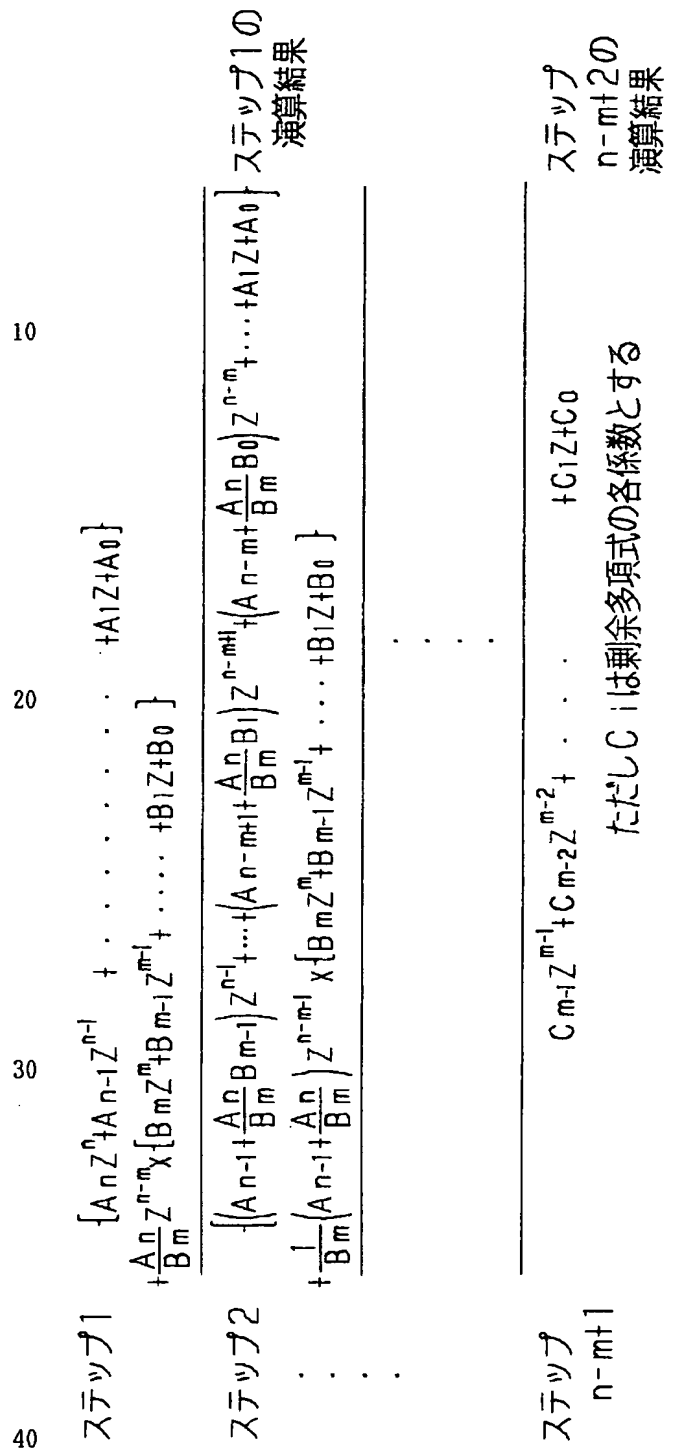
除多項式：$M2(Z)=B_m Z^m+B_{m-1} Z^{m-1}+\cdots B_1 Z+B_0$

とする。なお、$n>m$とする。以下、筆算により上記除算を行った場合の計算手順を数４に示す。なお、本実施例では初期値として、被除多項式の各係数データがレジ

スタ81a に、また除多項式の各係数データがレジスタ81 b に記憶されているものとする。数４に示すようにまず始めは除多項式の各係数データに除算回路100 より出力される除算結果を掛け合わせることにより除多項式の最大次数の係数データを被除多項式の最大次数の係数データと同一にして除算（被除多項式の次数を１次減らす動作）を行う。この除算を従来例と同様に被除多項式の剰余の次数が除多項式の次数よりも小さくなるまで繰り返す。なお、数４では最終の剰余多項式の各係数データを C_i （ｉ＝０，１，２，・・・・，ｍ－１）で表わす。

【０１５０】

【数４】



【０１５１】図25に示すユークリッドアルゴリズムに従うと上記剰余多項式

剰余多項式：$R(Z)=C_{m-1}Z^{m-1}+C_{m-2}Z^{m-2}+\cdots+C_1Z+C_0$

の次数が予め定められた条件を満たしていない場合は上記除多項式Ｍ２（Ｚ）を被除多項式として、また上記剰余多項式Ｒ（Ｚ）を除多項式として除算を繰り返す。

【０１５２】数５にＭ２（Ｚ）をＲ（Ｚ）で除算した場合の計算手順を示す。なお、本筆算では便宜上、上記剰余多項式Ｒ（Ｚ）の最大次数 C_{m-1} は０でないものとす

41

る。一般に、０であった場合は従来例と同様に図11に示す次数検出回路87，88で上記被除多項式及び除多項式の次数を判定して被除多項式の最大次数が０でなくなるまで被除多項式の係数データをシフトして除算を実行する。また、除算演算中の剰余多項式の最大次数が０になった場合も同様に剰余多項式の最大次数が０でなくなるまで剰余多項式の係数データをシフトして除算を行う。

【０１５３】

【数５】

$$\frac{C_{m-1}}{B_m}\left\{B_m Z^m + B_{m-1}Z^{m-1} + \cdots\cdots + B_2 Z^2 + B_1 Z + B_0\right\} \text{—被除数}$$

$$+Z\left\{C_{m-1}Z^{m-1} + C_{m-2}Z^{m-2} + \cdots\cdots + C_1 Z + C_0\right\} \text{—除数}$$

$$C_{m-1} \Big/ \left(\frac{C_{m-1}}{B_m}B_{m-1} + C_{m-2}\right) \times \left\{\left(\frac{C_{m-1}}{B_m}B_{m-1} + C_{m-2}\right)Z^{m-1} + \cdots + \left(\frac{C_{m-1}}{B_m}B_2 + C_1\right)Z^2 + \left(\frac{C_{m-1}}{B_m}B_1 + C_0\right)Z + \frac{C_{m-1}}{B_m}B_0\right\}$$

$$+\left\{C_{m-1}Z^{m-1} + C_{m-2}Z^{m-2} + \cdots\cdots + C_2 Z^2 + C_1 Z + C_0\right\}$$

$$E_{m-2}Z^{m-2} + E_{m-3}Z^{m-3} + \cdots\cdots + E_1 Z + E_0 \text{—剰余}$$

ただしＥｉは剰余多項式の各係数とする

ステップ1　ステップ2

【０１５４】なお、これらの制御はユークリッド演算制

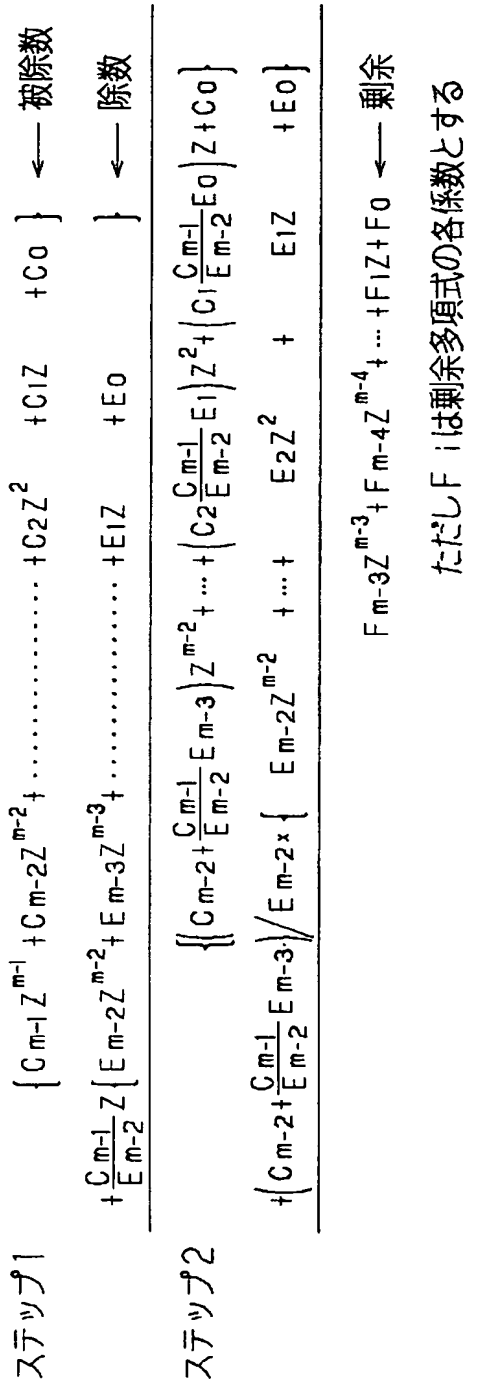
42

御回路101 で行う。図11に示す本実施例では、従来例で示したような被除多項式の係数データと除多項式の係数データとを交換するためのセレクタが配置されていない。従って、除算を実行する際は、数５に示すように被除多項式の各係数データに除算回路100 より出力される乗数を掛け合わせることにより被除多項式の最大次数の係数データを除多項式の最大次数の係数データと同一にして除算（被除多項式の次数を１次減らす動作）を行う。この除算を従来例と同様に被除多項式の剰余の次数が除多項式の次数よりも小さくなるまで繰り返す。その結果、剰余多項式（Ｒ”（Ｚ）とする）の各係数データをE_i（ｉ＝０，１，２，・・・，ｍ－２）で表わす。

【０１５５】そして、剰余多項式Ｒ”（Ｚ）（Ｒ”（Ｚ）＝$E_{m-2}Z^{m-2} + E_{m-3}Z^{m-3} + \cdots + E_1 Z + E_0$）の次数が図25に示した条件を満たしていない場合上記Ｒ（Ｚ）を被除多項式として、Ｒ”（Ｚ）を除多項式として除算を繰り返す。数６に筆算手順を示す。なお、この場合も説明を簡単にするため剰余多項式の最大次数の係数であるE_{m-2}は０でないものとする。先ほど述べたように被除多項式と除多項式の係数データの交換を行うためのセレクタが配置されていない。

【０１５６】

【数６】



【０１５７】従って、数４と同様に除算を実行する際は、除多項式の各係数データに除算回路100 より出力される乗数を掛け合わせることにより除多項式の最大次数の係数データを被除多項式の最大次数の係数データと同一にして除算（被除多項式の次数を１次減らす動作）を行う。

【０１５８】この除算を従来例と同様に被除多項式の剰余の次数が除多項式の次数よりも小さくなるまで繰り返す。その結果、剰余多項式（R■■■（Z）とする）の各係数データをF_i（i＝0，1，2，・・・，m－3）で表わす。そして、この動作をユークリッドアルゴリズムの終了条件を満たすまで実行する。なお、この動作の繰り返しで誤り数値多項式ω（Z）と誤り位置多項式σ（Z）とが算出される理由は、誤り位置または誤り数値を算出する際、上記多項式を各々の多項式の最大次数で正規化するためである。（詳しくは前述した「符号理論」，pp.169～175 を参照）

【０１５９】以下、図25に示すユークリッドアルゴリズムに従い、リード・ソロモン符号を復号する誤り訂正装置のユークリッド演算の除算回路部の動作を図11を用いて説明する。本実施例では、従来例と同様に光ディスク等に採用されている最小ハミング距離が17のリード・ソロモン符号を復号する場合について説明する。いま、再生信号より生成したシンドローム多項式Ｓ（Ｚ）を

$S(Z)=S_{15}Z^{15}+S_{14}Z^{14}+S_{13}Z^{13}+\cdots+S_1Z+S_0$

とする。

【０１６０】ユークリッド演算が開始されるとまず始め、ユークリッド演算制御回路101 より出力されるユークリッド演算開始信号をもとにレジスタ81a に初期値である被除多項式Z^{2t}の係数が記憶されるとともに、レジスタ81b に除多項式であるシンドローム多項式Ｓ（Ｚ）の各係数データが記憶される。また、次数検出回路87,88のカウンタにはそれぞれ次数の初期値である16と15とがセットされる。次数検出回路87，88内のカウンタは被除多項式の係数データを記憶しているレジスタ８１が１回シフトされるたびにカウント値が１つ下がるように構成されている。なお、本実施例では従来例と同様に右側に配置されているレジスタに高次の次数の係数データが記憶される。

【０１６１】それと同時に、除多項式であるシンドローム多項式は次数検出回路８８で最高次数の係数データが０であるか判定する。ユークリッド演算制御回路101 では、次数検出回路88で検出された除多項式（シンドローム多項式）の最大次数の係数が０の場合、除多項式の係数データが記憶されているレジスタ81b の内容をシフトし（なお、この際加算回路86での加算動作を行わずレジスタ81b の内容をそのままシフトする構成に成っている。）最大次数のデータが０でなくなるまでシフト動作を繰り返す。その際、次数検出回路88中のカウンタはカウント値をシフト動作が行われるたびに１づつ減らされる。

【０１６２】また、被除多項式の係数データが記憶されているレジスタ81a はクロックが入力されると加算回路86の出力がラッチされるように構成されており、除多項式の係数データが記憶されているレジスタ81b は係数データが保持されるように構成されている。（図11には、具体的な制御回路は省略したが、レジスタ81a，81bに入力されるクロック（クロック１及びクロック２）を除数側の係数データが記憶されているレジスタについては供給しないという方法で除多項式の係数データをレジスタ

内に保持する構成を取っている。）なお、図中に記した各演算回路（除算回路100,乗算回路85及び加算回路86）はガロア体上で定義された各演算を行うものとする。

【０１６３】以下、図12，13を用いて本実施例のユークリッド演算回路中の除算回路の回路動作を説明する。図12(a) にレジスタ81a に被除多項式として初期値であるZ^{16}がセットされ、81b に除多項式としてシンドローム多項式Ｓ（Ｚ）がセットされた状態を示す。なお、レジスタ中に記した数字、及び記号はレジスタに格納されている係数データを示すものとする。本実施例では説明を簡単にするためシンドローム多項式の最高次の係数は０でないものとする。また、レジスタに記憶される各多項式の高次の次数の係数データは右端に配置されるものとする。除算回路100 はレジスタ81a の出力を被除数、レジスタ81b の出力を除数として除算しその結果を出力する。このとき、被除数側の係数データが記憶されているレジスタ81a 側にクロック１が供給される。

【０１６４】図12，13に示す除算回路は図38，39で説明した除算回路と同一の構成をとっており、被除多項式の剰余の次数が、除多項式の次数より小さくなるところまでシフト動作を繰り返すことにより除算を実行する。除算の終了判定は次数検出回路87，88より出力される次数情報をもとにユークリッド演算制御回路101 で判定を行う。なお、除算演算実行中に剰余多項式の最大次数が次数検出回路87（または88）により０と検出されるとユークリッド演算制御回路101 では剰余多項式の係数データが記憶されているレジスタ81a （または81b ）の内容をそのまま１つシフトするように構成されている。その際、次数検出回路87（または88）中のカウンタのカウント値が１つ減らされる。

【０１６５】図12(b) に上記除算が終了した際に、各レジスタ81に記憶されている係数データの状態を示す。なお、レジスタ81a に記憶されているH_{14}～H_0 は上記除算を行った際の剰余多項式の係数データである。除算演算が終了すると、次数検出回路87（または88）では剰余多項式の最大次数を検出し最大次数が０であったなら、ユークリッド演算制御回路101 より出力される制御信号をもとに、剰余多項式の係数をそのまま１つシフトする。この動作を剰余多項式の最大次数の係数データが０でなくなるまで繰り返す。その際、上述のように、次数検出回路87（または88）中のカウンタのカウント値が１つづつ減らされる。

【０１６６】ユークリッド演算制御回路101 では、次数検出回路87（または88）より出力される上記剰余多項式の次数をもとに、剰余の次数がユークリッド演算の終了条件（本実施例では次数が７以下）を満たしていればユークリッド演算を終了し、終了条件を満たしていなければ前回の演算で除多項式であった多項式を被除多項式として、また上記剰余多項式を除多項式として演算を実行する。その際、レジスタ81に格納されている係数データを図13(a) に示す。従来例のように交換手段を有していないので回路の接続状態は変わらないが、レジスタ81a に供給されていたクロック１が止められて、反対にレジスタ81b にクロック２が供給される。よって、レジスタ81a の内容が保持され、レジスタ81b に剰余多項式の係数が逐次記憶される。

【０１６７】その際、ユークリッド演算回路中の除算回路では、次数検出回路88により被除多項式（剰余多項式）の最大次数が０であるかを判定し、０であった場合レジスタ81b に記憶されている被除多項式（剰余多項式）の内容をそのまま１つシフトするように制御信号を出力する。これは、除算回路100 へ出力される除数と被除数を格納するレジスタが決まっており被除多項式の係数データが除数として用いられる場合、被除多項式の最大次数の係数が０となってしまうと被除数（除多項式の最大係数）を０で割ることになり、解（出力）が不定と成ってしまう場合が生じるためである。

【０１６８】除算は、上記例と同様に被除多項式の剰余の次数が除多項式の次数より小さくなるまで被除多項式の係数データを記憶しているレジスタの内容を演算しながらシフト動作を繰り返すことにより除算を実行する。図13(b) に上記除算が終了した際に、各レジスタ81a，81bに記憶されている係数データの状態を示した。なお、レジスタ81b に記憶されているJ_{13}～J_0 は剰余多項式の係数データである。

【０１６９】そして、上記剰余多項式の次数を判定し所定の条件を満たしていなければレジスタ81a，81bに供給するクロックを切り換えてユークリッドアルゴリズムに基づく除算動作を再び行う。この除算動作を剰余多項式の次数が７次以下になるまで繰り返す。これにより、リード・ソロモン符号を復号する際に、ユークリッドアルゴリズムに基づくガロア体上の多項式同士の除算を１多項式毎に行うことができ、誤り訂正処理速度を向上させることが出来るとともに交換手段（セレクタ）を持たないため回路規模の縮小が図れ、またセレクタ制御も必要なくなる。

【０１７０】第６実施例．図14に本発明の第６実施例を示す。図14において回路構成は第５実施例（図11）と同様であるので詳細な説明は省略する。なお、第６実施例ではレジスタに接続されている乗算回路85の位置が第５実施例とは異なる。ユークリッド演算開始時に初期値であるZ^{2t}がセットされるレジスタ側に乗算回路85を接続した点が第５実施例とは異なる。この構成により、第５実施例に比べて乗算回路85を１つ省略でき回路規模の縮小（約 400ゲート程度の縮小）が図れる。

【０１７１】以下、第５実施例と同様に図25に示すユークリッドアルゴリズムに従い、リード・ソロモン符号を復号する誤り訂正装置のユークリッド演算の除算回路部の動作を図14を用いて説明する。第６実施例では、第５実施例と同様に光ディスク等に採用されている最小ハミ

ング距離が17のリード・ソロモン符号を復号する場合について説明する。いま、再生信号より生成したシンドローム多項式S（Z）を

$S(Z)=S_{15}Z^{15}+S_{14}Z^{14}+S_{13}Z^{13}+\cdots+S_1Z+S_0$

とする。

【0172】ユークリッド演算が開始されるとまず始め、ユークリッド演算制御回路101 より出力されるユークリッド演算開始信号をもとにレジスタ81a に初期値である被除多項式Z^{2t}の係数が記憶されるとともに、レジスタ81b に除多項式であるシンドローム多項式S（Z）の各係数データが記憶される。また、次数検出回路87,88のカウンタにはそれぞれ次数の初期値である15と16とがセットされる。次数検出回路87，88内のカウンタは被除多項式の係数データを記憶しているレジスタ81a，81bが1回シフトされるたびにカウント値が1つ下がるように構成されている。

【0173】それと同時に、除多項式であるシンドローム多項式は次数検出回路88で最高次数の係数データが0であるか判定する。ユークリッド演算制御回路101 では、次数検出回路88で検出された除多項式（シンドローム多項式）の最大次数の係数データが0の場合、除多項式の係数データが記憶されているレジスタ81b の内容をシフトし（なお、この際加算回路86での加算動作を行わずレジスタ81b の内容をそのままシフトする構成になっている。）最大次数のデータが0でなくなるまでシフト動作を繰り返す。その際、次数検出回路88中のカウンタはカウント値を1づつ減らされる。

【0174】また、被除多項式の係数データが記憶されているレジスタ81a はクロックが入力されると加算回路86の出力がラッチされるように構成されており、除多項式の係数データが記憶されているレジスタ81b は係数データが保持されるように構成されている。図14では、具体的な制御回路は省略したが、第5実施例と同様にレジスタ81a，81bに入力されるクロック1，2をコントロールすることにより除多項式の係数データをレジスタ内に保持する構成を取っている。なお、図中に記した各演算回路はガロア体上で定義された各演算を行うものとする。

【0175】以下、図15,16,17を用いて本実施例のユークリッド演算回路中の除算回路の回路動作を説明する。図15(a) にレジスタ81a に被除多項式として初期値であるZ^{16}がセットされ、レジスタ81b に除多項式としてシンドローム多項式S（Z）がセットされた状態を示す。なお、レジスタ中に記した数字、及び記号はレジスタに格納されている係数データを示すものとする。本実施例では説明を簡単にするためシンドローム多項式の最高次の係数は0でないものとする。また、レジスタに記憶される各多項式の高次の次数の係数データは右端に配置されるものとする。除算回路100 はレジスタ81a の出力を

除数、レジスタ81b の出力を被除数として除算しその結果を出力する。このとき、除数側の係数データが記憶されているレジスタ81a 側にクロック1が供給される。

【0176】図15(b) に図15(a) に示す状態より1回レジスタ81a 側をシフトした状態を示した。被除数側の初期値が図25に示すユークリッドアルゴリズムに示すようにZ^{16}であるので被除多項式の最小次数の係数が0となる。よって、図14に示す回路構成の場合、1回目の除算によって求められる被除多項式側の剰余多項式の最小次数の係数データは

$S_0+0\times S_{15}=S_0$

となり、乗算回路85を省略できる事がわかる。図15(b) に上記1回目の除算後のレジスタ81a，81bに記憶されている各係数データを示す。図中に記したK_{14}～K_1 及びS_0 は剰余多項式の各係数データである。

【0177】第5実施例同様、被除多項式の剰余の次数が、除多項式の次数より小さくなるところまでシフト動作を繰り返すことにより除算を実行する。除算の終了判定は次数検出回路87，88より出力される次数情報をもとにユークリッド演算制御回路101 で判定を行う。なお、除算演算実行中に剰余多項式の最大次数が次数検出回路87（または88）により0と検出されるとユークリッド演算制御回路101 では剰余多項式の係数データが記憶されているレジスタ81a （または81b ）の内容をそのまま1つシフトするように構成されている。その際、次数検出回路87（または88）中の次数カウンタのカウント値が1つ減らされる。これは、除算回路100 へ出力される除数と被除数を格納するレジスタが決まっており被除多項式の係数データが除数として用いられる場合、被除多項式の最大次数の係数が0となってしまうと被除数（除多項式の最大係数）を0で割ることになり、解（出力）が不定と成ってしまう場合が生じるためである。

【0178】図16(a) に上記除算が終了した際に、各レジスタ81a，81bに記憶されている係数データの状態を示す。なお、レジスタ81a に記憶されているL_{14}～L_0 は上記除算を行った際の剰余多項式の係数データである。なお、本実施例では$L_0=S_0$ である。除算演算が終了すると、次数検出回路87（または88）では剰余多項式の最大次数を検出し最大次数が0であったなら、ユークリッド演算制御回路101より出力される制御信号をもとに、剰余多項式の係数をそのまま1つシフトする。この動作を剰余多項式の最大次数の係数データが0でなくなるまで繰り返す。その際、上述のように、次数検出回路87（または88）中の次数カウンタのカウント値が1つづつ減らされる。

【0179】ユークリッド演算制御回路101 では、次数検出回路87（または88）より出力される上記剰余多項式の次数をもとに、剰余の次数がユークリッド演算の終了条件（本実施例では次数が7以下）を満たしていればユークリッド演算を終了し、終了条件を満たしていなけれ

ば前回の演算で除多項式であった多項式を被除多項式として、また上記剰余多項式を除多項式として演算を実行する。その際、レジスタ81a，81bに格納されている係数データを図16(b) に示す。従来例と同様に交換手段を有していないので回路の接続状態は変わらないが、レジスタ81a に供給されていたクロック１が止められて、反対にレジスタ81b にクロック２が供給される。よって、レジスタ81a の内容が保持され、レジスタ81b に剰余多項式の係数が逐次記憶される。

【０１８０】その際、ユークリッド演算回路中の除算回路では，次数検出回路88により被除多項式（剰余多項式）の最大次数が０であるかを判定し、０であった場合レジスタ81b に記憶されている被除多項式（剰余多項式）の内容をそのまま１つシフトするように制御信号を出力する。

【０１８１】除算は、第５実施例と同様に被除多項式の剰余の次数が除多項式の次数より小さくなるまで被除多項式の係数データを記憶しているレジスタの内容を演算しながらシフト動作を繰り返すことにより除算を実行する。図17に上記除算が終了した際に、各レジスタ81a，81bに記憶されている係数データの状態を示す。なお、レジスタ81b に記憶されているM_{13}～M_0 は剰余多項式の係数データである。

【０１８２】そして、上記剰余多項式の次数を判定し所定の条件を満たしていなければレジスタ81a，81bに供給するクロックを切り換えてユークリッドアルゴリズムに基づく除算動作を再び行う。この除算動作を剰余多項式の次数が７次以下になるまで繰り返す。これにより、リード・ソロモン符号を復号する際に、ユークリッドアルゴリズムに基づくガロア体上の多項式同士の除算を１多項式毎に行うことができ、誤り訂正処理速度を向上させることが出来るとともに交換手段（セレクタ）を持たないため回路規模の縮小が図れ、またセレクタ制御も必要なくなる。また、第５実施例に比べて乗算回路85を１つ省略でき回路規模の縮小を図れる。

【０１８３】第７実施例．本発明の第７実施例は、第５，第６実施例で用いられた加算回路86及び除算回路100 の構成とユークリッド演算回路101 の制御方法とが異なる。よって、第７実施例は図11に示すユークリッド演算の除算回路を用いて説明する。図18に除算回路100 の具体的な構成図を示す。110 は入力された係数の逆数を出力する逆元ＲＯＭ、111 は乗算回路、112 はＯＲゲートである。逆元ＲＯＭ110 は入力されたガロア体上の元の逆数を出力するように構成されている。図19には加算回路86の構成を示す。図において、113 はＡＮＤゲート、114 はＥＸＯＲゲートである。表１に本実施例の逆元ＲＯＭ110 の内容を示す。なお、表中に記したαは原始多項式Ｇ（Ｚ）のガロア体上の元である。表に示すように本実施例における逆元ＲＯＭ110 は０が入力されると０が出力される構成になっている。また、表１に記したαはＧＦ（2^8）で定義されるガロア体上の元である。

【０１８４】

【表１】

逆元ＲＯＭの内容

入力データ	出力データ
0	0
1	1
α^1	α^{254}
α^2	α^{253}
α^3	α^{252}
α^4	α^{251}
α^5	α^{250}
・	・
・	・
・	・
α^{250}	α^5
α^{251}	α^4
α^{252}	α^3
α^{253}	α^2
α^{254}	α^1

【０１８５】以下、図11を用いて第７実施例の回路動作を説明する。第５実施例と同様、再生信号より生成したシンドローム多項式Ｓ（Ｚ）を

$S(Z)=S_{15}Z^{15}+S_{14}Z^{14}+S_{13}Z^{13}+\cdots+S_1Z+S_0$

とする。

【０１８６】ユークリッド演算が開始されるとまず始め、ユークリッド演算制御回路101 より出力されるユークリッド演算開始信号をもとにレジスタ81a に初期値である被除多項式Z^{2t}の係数が記憶されるとともに、レジスタ81b に除多項式であるシンドローム多項式Ｓ（Ｚ）の係数データが記憶される。また、次数検出回路87，88のカウンタにはそれぞれ次数の初期値である16と15とがセットされる。

【０１８７】それと同時に、除多項式であるシンドロー

ム多項式は次数検出回路88で最高次数の係数データが検出される。ユークリッド演算制御回路101 では、次数検出回路88で検出された除多項式（シンドローム多項式）の次数が０の場合、除多項式の係数データが記憶されているレジスタ81b の内容をシフトし最大次数のデータが０でなくなるまでシフト動作を繰り返す。その際、次数検出回路88中の次数カウンタのカウント値は１づつ減らされる。

【０１８８】この時、除算回路100 では除数側のデータとして０が入力される。逆元ＲＯＭ110 では表１に示すように０が入力されると０が出力される構成になっている。逆元ＲＯＭ110 により出力されたデータは乗算回路111 で被除数と乗算され出力される。この場合、逆元ＲＯＭ110 からの出力データが０であるので乗算回路111 の出力は０となる。ＯＲゲート112 は乗算回路111 より出力されるＬＳＢデータとユークリッド演算制御回路101 より出力される制御信号との論理和をとる。ユークリッド演算制御回路101 では次数検出回路88で検出された多項式の最大次数の係数データが０である場合に”Ｈ”を出力する構成になっている。よって、除算回路100 からは１が出力されることになる。

【０１８９】また、加算回路86中のＡＮＤゲート113 の一方の入力端子には、除算回路100の被除数側の係数データが格納されているレジスタ81a より出力される係数データの各ビットが入力されるように構成されており、他方の入力には、ユークリッド演算制御回路101 より出力される制御信号が入力される。この制御信号は、除算回路100 の除数側の係数データが次数検出回路88で０と検出された場合”Ｌ”を出力するように構成されている。この構成により、除算回路100 へ入力される除数側（レジスタ81b 側）の係数データは

$$1 \times S_j + 0 = S_j \qquad (j = 0, 1, \cdots, 15)$$

の演算が行われシフト動作が繰り返されることになる。除数多項式の最大次数の係数データが０でなくなるまでシフト動作を繰り返す。これにより、上記係数データをシフトする際ユークリッド演算制御回路101 では特別な制御を必要とせず、係数シフトを行うレジスタさえ意識していればよいことになる。

【０１９０】シンドローム多項式の最大次数が検出されるとユークリッドアルゴリズムに基づく除算が開始される。第５実施例と同様に、被除多項式の係数データが記憶されているレジスタ81a はクロックが入力されると加算回路86の出力がラッチされるように構成されており、除多項式の係数データが記憶されているレジスタ81bは係数データが保持されるように構成されている。

【０１９１】除算動作は、被除多項式の剰余の次数が、除多項式の次数より小さくなるまで繰り返される。除算の終了判定は次数検出回路87，88より出力される次数情報をもとにユークリッド演算制御回路101 で判定を行う。なお、除算演算実行中に剰余多項式の最大次数が次数検出回路87により０と検出された場合でも除算回路100 より０が出力されるので乗算回路85及び加算回路86では

$$0 \times S_i + X_i = X_i$$

の計算が行われレジスタ81a にはX_i（X_i はレジスタ81a に記憶されている係数データ）がそのままシフトされ記憶されることになる。従って、ユークリッド演算制御回路101 では特に制御を変えることなくシフト動作を繰り返せばよい。

【０１９２】除算演算が終了すると、次数検出回路87（または88）では剰余多項式の最大次数を検出し最大次数が０であったなら、レジスタ81a の内容をそのままシフトする。その際、上述のように、次数検出回路87（または88）中のカウンタのカウント値が１つづつ減らされる。

【０１９３】ユークリッド演算制御回路101 では、次数検出回路87（または88）より出力される上記剰余多項式の次数をもとに、剰余の次数がユークリッド演算の終了条件（本実施例では次数が７以下）を満たしていればユークリッド演算を終了し、終了条件を満たしていなければ前回の演算で除多項式であった多項式を被除多項式として、また上記剰余多項式を除多項式として演算を実行する。その際、従来例のように交換手段を有していないので回路の接続状態は変わらないが、レジスタ81a に供給されていたクロック１が止められて、反対にレジスタ81b にクロック２が供給される。よって、レジスタ81a の内容が保持され、レジスタ81b に剰余多項式の係数が逐次記憶される。

【０１９４】この時、ユークリッド演算回路中の除算回路では、次数検出回路88により被除多項式（剰余多項式）の最大次数（レジスタ81b の出力）が０であった場合でも上述で述べた通りユークリッド演算制御回路101 では各加算回路86及び除算回路100 にそれぞれ”Ｌ”及び”Ｈ”の制御信号を出力するだけでレジスタ81b の内容がシフトされる構成になっており、０を意識した制御を行わなくてもよい。すなわち、次数検出回路88によりレジスタ81b に記憶されている多項式の最大次の係数データが０と判断されると除算回路100 からは除算結果として１が出力される。一方、加算回路86ではユークリッド演算制御回路101 より出力される制御信号によりＡＮＤゲート113 の出力が”Ｌ”となる。（すなわち、加算回路86に０が入力されることになる。）従って、乗算回路85及び加算回路86では

$$1 \times Y_i + 0 = Y_i$$

の演算が実行される。よって、レジスタ81b にはＹ$_i$（Y_i はレジスタ81b に記憶されている係数データ）がそのままシフトされ記憶されることになる。

【０１９５】除算は、上記例と同様に被除多項式の剰余の次数が除多項式の次数より小さくなるまで被除多項式

の係数データを記憶しているレジスタの内容を演算しながらシフト動作を繰り返すことにより除算を実行する。そして、上記剰余多項式の次数を判定し所定の条件を満たしていなければレジスタ81a，81bに供給するクロックを切り換えてユークリッドアルゴリズムに基づく除算動作を再び行う。この除算動作を剰余多項式の次数が７次以下になるまで繰り返す。これにより、リード・ソロモン符号を復号する際に、ユークリッドアルゴリズムに基づくガロア体上の多項式同士の除算を１多項式毎に行うことができ、誤り訂正処理速度を向上させることが出来るとともにユークリッド演算制御回路101 での制御も非常に簡単になり回路規模の削減を図れる。

【０１９６】なお、第７実施例では除算回路100 を逆元ＲＯＭ110 及び乗算回路111 で構成したがこれに限るものではなく、入力された除数が０である場合除算回路100 の出力を１と出力するように除算回路100 を構成すればロジックゲート、またはマイクロコンピュータ等で構成しても同様の効果を奏する。また、第７実施例では入力されたデータの逆元を求める際、ＲＯＭを使用したがＲＡＭでもよく、またロジックゲートで逆元を求めるように構成してもよいことはいうまでもない。また、加算回路86の構成は図19に示す構成に限るものではなく、ユークリッド演算回路101 より出力される制御信号に基づきレジスタ81b の出力をそのまま出力するように構成すれば同様の効果を奏する。また、ＧＦ（2^8）で定義される場合について述べたがこれに限るものではない。ＧＦ（ｑ）（ｑは素数、または素数のべき表現）で定義されるガロア体上の演算なら同様の効果を奏する。更に、図11に示す除算回路の場合について述べたが図14に示す回路構成でも同様の効果を奏する。また、第７実施例は従来例に示す除算回路84及び加算回路86に適用しても同様にユークリッド演算制御回路89の制御を簡略化できることは言うまでもない。初期値を設定した際、シンドローム多項式の最大次の係数データが０であった場合、レジスタ81b 内のデータを最大次の係数データが０でなくなるまでシフトする時の回路構成及び制御を簡略化できる。

【０１９７】第８実施例．以下、本発明の第８実施例について説明する。本実施例では、図11に示す誤り訂正装置で除算回路100 の構成を図18に示す構成のものを用いた場合について説明する。図20は除算回路100 及びユークリッド演算制御回路101 より出力される除算結果、及び制御信号を用いて誤り位置多項式を算出する誤り位置算出回路の一部を示す。図において120 は逆元ＲＯＭ、121 は乗算回路、122 はセレクタである。なお、第８実施例では説明を簡単にするためガロア体上の除算を実行する際、被除多項式と除多項式の次数差が１次の場合について説明する。

【０１９８】図11に示す誤り訂正装置で多項式同士の除算を行った場合、商として除算回路100 より出力されるデータは次の２通りのパターンがある。一方は、被除多項式の係数データがレジスタ81a に記憶されている場合、もう一方は、被除多項式の係数データがレジスタ81b に記憶されている場合である。今、被除多項式及び除多項式をそれぞれ、

被除多項式：$M1(Z)=A_n Z^n+A_{n-1} Z^{n-1}+\cdots+A_1 Z+A_0$

除多項式：$M2(Z)=B_{n-1} Z^{n-1}+B_{n-2} Z^{n-2}+\cdots+B_1 Z+B_0$

とする。数７に被除多項式の係数データがレジスタ81a に記憶されている場合の除算結果を、数８に被除多項式の係数データがレジスタ81b に記憶されている場合の除算結果を示す。

【０１９９】

【数７】

【0200】

$$
\begin{array}{l}
\dfrac{A_n}{B_{n-1}}Z+\dfrac{1}{B_{n-1}}\left(A_{n-1}+\dfrac{B_{n-2}}{B_{n-1}}A_n\right) \\
\hline
B_{n-1}Z^{n-1}+B_{n-2}Z^{n-2}+\cdots+B_1Z+B_0 \;\Big|\; A_nZ^n+A_{n-1}Z^{n-1}+\cdots\cdots\cdots A_1Z+A_0 \\
\quad A_nZ^n+\dfrac{B_{n-2}}{B_{n-1}}A_nZ^{n-1}+\cdots+\dfrac{B_0}{B_{n-1}}A_nZ \\
\hline
\quad \left(A_{n-1}+\dfrac{B_{n-2}}{B_{n-1}}A_n\right)Z^{n-1}+\cdots+\left(A_1+\dfrac{B_0}{B_{n-1}}A_n\right)Z+A_0 \\
\quad \left(A_{n-1}+\dfrac{B_{n-2}}{B_{n-1}}A_n\right)Z^{n-1}+\cdots+\dfrac{B_1}{B_{n-1}}\left(A_{n-1}+\dfrac{B_{n-2}}{B_{n-1}}A_n\right)Z+\dfrac{B_1}{B_{n-1}}\left(A_{n-1}+\dfrac{B_{n-2}}{B_{n-1}}A_n\right) \\
\hline
\quad P_{n-2}Z^{n-2}+\cdots\cdots\cdots\cdots\cdots+P_1Z+P_0
\end{array}
$$

P_iは（$i=0\sim n-2$）剰余多項式の各係数

⇩

商多項式は $\dfrac{A_n}{B_{n-1}}Z+\dfrac{A_{n-1}B_{n-1}+B_{n-2}A_n}{B_{n-1}^2}$

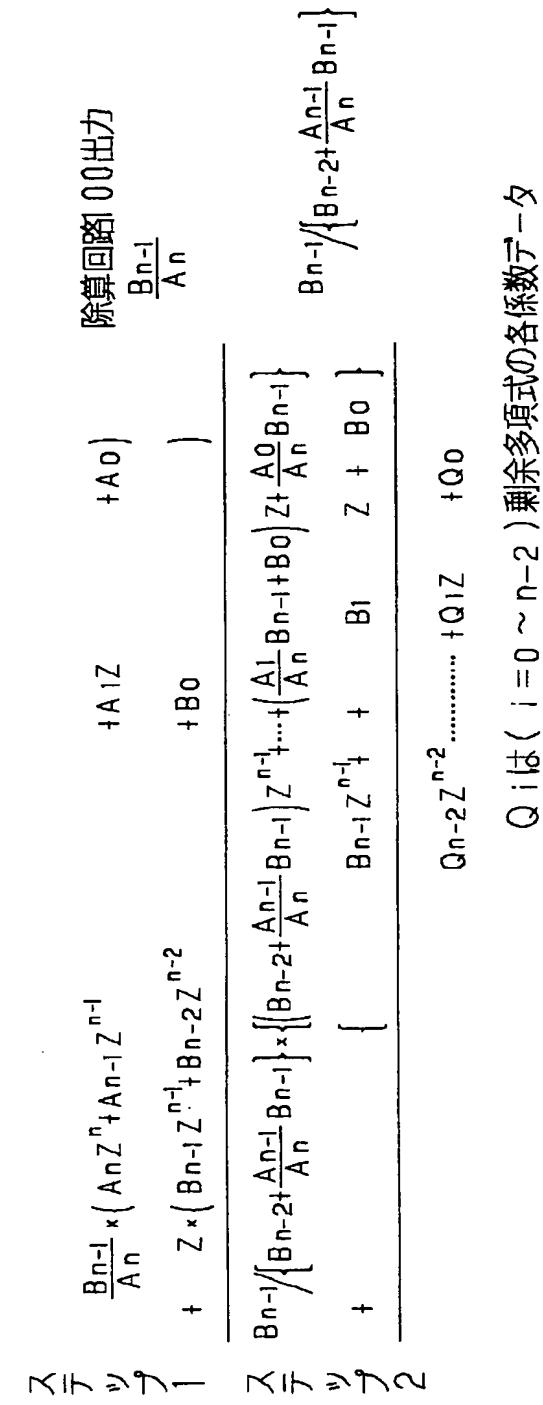
⇩

よって、誤り位置多項式の係数データを求める際は

$$\frac{A_nB_{n-1}}{A_{n-1}B_{n-1}+B_{n-2}A_n}Z+1$$

【数8】

57　　58



【０２０１】被除多項式の係数データがレジスタ81a に記憶されている場合、除算は数７に示すように実行され、誤り位置多項式の係数データを算出する際は０次の項を正規化（正規化して１にする）した後、図25に示すアルゴリズムに従い誤り位置多項式の係数データを求める。この場合、除算回路100 より出力される除算結果はそのまま商多項式の係数データに一致する。その結果を正規化した場合の商多項式を数７の一番下の式に示す。同様に、被除多項式の係数データがレジスタ81bに記憶されている場合について数８を用いて説明する。数８には各ステップにおける演算過程を示す。右側には各々のステップにおける除算回路100 の出力を示す。この場合、除算回路100 より出力される係数データは商多項式の係数データとは一致しない。しかしながら、数８に示す計算方法の場合、誤り位置多項式の係数データを算出する際の正規化した式を求める場合商多項式が１次の多項式の場合はステップ２の除算回路100 の出力がそのまま商多項式の１次の係数になっていることが数７と比較するとわかる。これをもとに図20に示す回路の動作を説明する。

【０２０２】図20において、逆元ＲＯＭ120 には、図11に示す除算回路100 より出力される０次の商多項式の係

数が入力される。また、乗算回路121 の一方の入力には逆元ＲＯＭ121 の出力が、もう一方の入力には除算回路100 より出力される１次の商多項式の係数が入力される。セレクタ122 では、ユークリッド演算制御回路101 より出力される制御信号に基づき、被除多項式の係数がレジスタ81a に記憶されている場合は乗算回路121 の出力を選択し、被除多項式の係数データがレジスタ81b に記憶されている場合は０次の商多項式の係数データが出力されるように構成されている。

【０２０３】この構成によると、レジスタ81a に被除多項式の係数データが記憶されている場合（数７に相当する）は除算結果が、誤り位置多項式を算出する際に用いられる商多項式の１次の係数となり、レジスタ81b に被除多項式の係数データが記憶されている場合（数８に相当する）は、除算回路100 より出力される０次の係数データが上記商多項式の１次の係数データとして出力される。これにより、商多項式が正規化され、誤り位置多項式を算出する際、セレクタ122 を追加するのみの簡単な回路構成で誤り位置多項式を算出できる。また、数８に示すように、レジスタ81b に被除多項式の係数データが記憶されている場合は、除算演算を行う事なく係数データが算出されるので復号ステップ数の削減が図れる。

【０２０４】なお、本実施例では商多項式として１次のものについて扱ったが、２次または３次等の場合についても回路構成は少し複雑になるが、簡単な回路構成で実現できる。以下、２次以上の場合について簡単に説明する。表２に被除多項式及び除多項式をそれぞれ、

被除多項式：$M1(Z)=A_n Z^n+A_{n-1} Z^{n-1}+\cdots+A_1 Z+A_0$

除多項式：$M2(Z)=B_{n-2} Z^{n-2}+B_{n-3} Z^{n-3}+\cdots+B_1 Z+B_0$

とした場合に、被除多項式の係数データがレジスタ81a に記憶されている場合（除多項式の最大次数の係数データが除算回路100 において除数として扱われている場合、従来例に示す通常の除算の場合）の除算結果と、被除多項式の係数データがレジスタ81b に記憶されている場合（除多項式の最大次数の係数データが除算回路100 において被除数として扱われている場合）の計算結果を示す。なお、本実施例においてもユークリッド演算を行なう際の除算回路部の構成は図11に示す回路構成を採用するものとする。

【０２０５】

【表２】

商次数	2次	1次	0次
除多項式が除算回路100の除数側の場合	$\frac{A_n}{B_{n-2}}$	$\frac{A_{n-1}B_{n-2}+A_nB_{n-3}}{(B_{n-2})^2}$	$\frac{B_{n-2}\{A_{n-2}B_{n-2}+A_nB_{n-4}\}+B_{n-3}\{A_{n-1}B_{n-2}+A_nB_{n-3}}{\{b_{n-2}\}^3}$
除多項式が除算回路100の被除数側の場合	$\frac{B_{n-2}}{A_n}$	$\frac{A_nB_{n-2}}{A_{n-1}B_{n-2}+A_nB_{n-3}}$	$\frac{B_{n-2}\{A_{n-1}B_{n-2}+A_nB_{n-3}\}}{B_{n-2}\{A_{n-2}B_{n-2}+A_nB_{n-4}\}+B_{n-3}\{A_{n-1}B_{n-2}+A_nB_{n-3}}$

【０２０６】数９には上記通常の除算を行った商多項式を正規化した場合の係数データを示す。これより、被除多項式の係数データがレジスタ81b に記憶されている場合（除多項式の最大次数の係数データが除算回路100 において被除数として扱われている場合）に、上記正規化した商多項式を求める際は、除算回路100 より出力される商データをP_i（ｉ＝０，１，・・・，ｋ；ｋは商多項式の次数）とし、商多項式の係数データをQ_i（ｉ＝０，１，・・・，ｋ；ｋは商多項式の次数）とすると、

$Q_0=1$

$Q_1=Q_0\times P_0$

61

$Q_2 = Q_1 \times P_1$

・

・

$Q_k = Q_{k-1} \times P_{k-1}$

となる。数10に、商多項式が２次の場合の最大次数のデ＊

62

＊ータを計算した結果を示す。（除多項式の最大次数の係数データが除算回路100 において被除数として扱われている場合）

【０２０７】

【数９】

$$\frac{A_n B_{n-2}^{\,2}}{B_{n-2}\{A_{n-2} B_{n-2} + A_n B_{n-4}\} + B_{n-3}\{A_{n-1} B_{n-2} + A_n B_{n-3}\}} x^2 + \frac{B_{n-2}\{A_{n-1} B_{n-2} + A_n B_{n-3}\}}{B_{n-2}\{A_{n-2} B_{n-2} + A_n B_{n-4}\} + B_{n-3}\{A_{n-1} B_{n-2} + A_n B_{n-3}\}} x + 1$$

【０２０８】

※ ※【数１０】

$$\frac{A_n B_{n-2}}{A_{n-1} B_{n-2} + A_n B_{n-3}} \times \frac{B_{n-2}\{A_{n-1} B_{n-2} + A_n B_{n-3}\}}{B_{n-2}\{A_{n-2} B_{n-2} + A_n B_{n-4}\} + B_{n-3}\{A_{n-1} B_{n-2} + A_n B_{n-3}\}} = \frac{A_n B_{n-2}^{\,2}}{B_{n-2}\{A_{n-2} B_{n-2} + A_n B_{n-4}\} + B_{n-3}\{A_{n-1} B_{n-2} + A_n B_{n-3}\}}$$

【０２０９】図21に上記商多項式が２次以上の場合についても考慮した、誤り位置多項式を算出する誤り位置算出回路の一部を示す。120 ～122 は図20に示すものと同一であるので詳細は省略する。123 はシフトレジスタで除算回路100 より出力される商多項式の各係数データ（除算結果）が次数の低い順に記憶される。124 はセレクタ、125 はレジスタである。逆元ＲＯＭ120 、乗算回路121 、セレクタ122，124、シフトレジスタ123 及びレジスタ125 で誤り位置算出回路の一部は構成されている。

【０２１０】次に、動作について説明する。図21において、逆元ＲＯＭ120 には、図11に示す除算回路100 より出力される０次の商多項式の係数が入力される。また、シフトレジスタ123 には図11に示す除算回路100 より出力される１次以上の商多項式の係数データが次数の低い順に逐次記憶される。セレクタ124 は逆元ＲＯＭ120の出力とレジスタ125 の出力をユークリッド演算回路101 より出力される制御信号に基づき切り換える。乗算回路121 ではシフトレジスタ123 の出力とセレクタ124 の出力を乗算する。なお、シフトレジスタ123 は乗算回路121 で乗算が実施されるたびにデータを逐次シフトする構成になっている。

【０２１１】セレクタ122 では、ユークリッド演算回路100 より出力される制御信号に基づき、被除多項式の係数がレジスタ81a に記憶されている場合は乗算回路121 の出力を選択し、被除多項式の係数データがレジスタ81b に記憶されている場合は、まづ初め０次の商多項式の係数データがそのまま出力され、商多項式の２次以降の係数を算出する際は乗算回路121 の出力を選択するように制御するように構成されている。セレクタ122 の出力はレジスタ125 に記憶される。セレクタ124 は図11に示すレジスタ81a が被除多項式として扱われている場合は逆元ＲＯＭ120の出力を選択し、レジスタ81b が被除多項式として扱われている場合はレジスタ125 の出力を選択するように構成されている。

＊【０２１２】この構成によると、レジスタ81a に被除多項式の係数データが記憶されている場合は除算回路100 より出力される各商多項式の係数データを０次の係数データで正規化された除算結果が、誤り位置多項式を算出する際に用いられる商多項式の係数データとなり、レジスタ81b に被除多項式の係数データが記憶されている場合は、セレクタ122，124を上述のように制御することにより上述の演算を実行し商多項式の係数データを算出することが可能となる。よって、セレクタ122，124を追加するのみの簡単な回路構成で誤り位置多項式を算出できる。

【０２１３】なお、上記実施例では商多項式として２次または３次等の場合について述べたが、この場合商多項式の各係数データに０が含まれている場合がある。このような場合も少し説明は複雑になるが、簡単な回路構成で実現できる。以下、２次の商多項式中１次の項が０であった場合について簡単に説明する。表３に被除多項式、及び除多項式をそれぞれ、

被除多項式：$M1(Z)=A_n Z^n+A_{n-1} Z^{n-1}+\cdots+A_1 Z+A_0$

除多項式：$M2(Z)=B_{n-2} Z^{n-2}+B_{n-3} Z^{n-3}+\cdots+B_1 Z+B_0$

とした場合に、被除多項式の係数データがレジスタ81a に記憶されている場合（除多項式の最大次数の係数データが除算回路100 において除数として扱われている場合、従来例に示す通常の除算の場合）の除算結果と、被除多項式の係数データがレジスタ81b に記憶されている場合（除多項式の最大次数の係数データが除算回路100 において被除数として扱われている場合）の計算結果とを示す。なお、本実施例においてもユークリッド演算を行なう際の除算回路部の構成を図11に示す回路構成を採用するものとする。また、本実施例を説明するため、商多項式の１次の項を０とする。

【０２１４】

＊【表３】

	２次	１次	０次
除多項式が除算回路 100の除数側の場合	$\frac{A_n}{B_{n-2}}$	0	$\frac{A_{n-2} B_{n-2}+B_{n-4} A_n}{\{B_{n-2}\}^2}$
除多項式が除算回路 100の被除数側の場合	$\frac{B_{n-2}}{A_n}$	0	$\frac{A_n B_{n-2}}{A_{n-2} B_{n-2}+B_{n-4} A_n}$

ただし$A_{n-1} B_{n-2}+A_n B_{n-3}=0$とする

【０２１５】数11には上記通常の除算を行った商多項式　50　を正規化した場合の係数データを示す。

65 66

【0216】

＊ ＊【数11】

商多項式

$$\frac{A_n B_{n-2}}{A_{n-2} B_{n-2} + B_{n-4} A_n} x^2 + 1$$

【0217】これより、被除多項式の係数データがレジスタ81b に記憶されている場合（除多項式の最大次数の係数データが除算回路100 において被除数として扱われている場合）に、上記正規化した商多項式を求める際は、除算回路100 より出力される商データをP_i（i＝0，1，・・・，k；kは商多項式の次数）とし、商多項式の係数データをQ_i（i＝0，1，・・・，k；kは商多項式の次数）とし、またP_j～P_m（0＜j≦m＜k）のデータを0とすると、

$Q_0 = 1$

$Q_1 = Q_0 \times P_0$

$Q_2 = Q_1 \times P_1$

・

・

$Q_j = 0$

・

・

$Q_{m+1} = Q_{j-1} \times P_{j-1}$

・

・

$Q_k = Q_{k-1} \times P_{k-1}$

となる。

【0218】図22に上記商多項式の係数中に0が存在した場合にも対応できる誤り位置多項式を算出する誤り位置算出回路の一部の回路構成を示す。120 ～125 は図21に示すものと同一であるので詳細は省略する。126 はレジスタ81b に記憶されている係数データが被除多項式の時に、次数検出回路88で検出された最大次の係数データが0であるかを識別するフラグデータを記憶するシフトレジスタ、128 はシフトレジスタ126 の出力をラッチしてデータとのタイミングを合わせるレジスタ、127 はレジスタ128 より出力される信号をもとにレジスタ125 の出力と0とを選択するセレクタである。逆元ＲＯＭ120 、乗算回路121 、セレクタ122,124,127、シフトレジスタ123，126及びレジスタ125，128で誤り位置算出回路の一部は構成されている。なお、本実施例で説明に用いる図11に示す除算回路100 の構成は図18に示す構成を取っているものとする。すなわち、除算回路100 の除数側のデータに0が入力された場合除算回路100 からは1が出力されるものとする。

【0219】次に、動作について説明する。図22において、逆元ＲＯＭ120 には、図11に示す除算回路100 より出力される0次の商多項式の係数が入力される。また、シフトレジスタ123 には図11に示す除算回路100 より出力される1次以上の商多項式の係数データが次数の低い順に逐次記憶される。なお、それと同時にシフトレジスタ126 にレジスタ81b が被除多項式として扱われる際、次数検出回路88で検出された0検出フラグ（本実施例では、0を検出すると”H”となるものとする）が記憶される。なお、0検出フラグデータはレジスタ81b に記憶されている係数データが被除多項式として用いられている場合、被除多項式の最大次数の係数が0であった場合出力される。セレクタ124 は逆元ＲＯＭ120 の出力とレジスタ125 の出力とをユークリッド演算制御回路101 より出力される制御信号に基づき切り換える。乗算回路121 ではシフトレジスタ123 の出力とセレクタ124 の出力とを乗算する。

【0220】なお、シフトレジスタ123，126は乗算回路121 で乗算が実施されるたびにデータを逐次シフトする構成になっている。セレクタ122 では、ユークリッド演算制御回路101 より出力される制御信号に基づき、被除多項式の係数データがレジスタ81a に記憶されている場合は乗算回路121 の出力を選択し、被除多項式の係数データがレジスタ81b に記憶されている場合は、まず初め0次の商多項式の係数データがそのまま出力され、商多項式の2次以降の係数を算出する際は乗算回路121 の出力を選択するように制御するように構成されている。セレクタ122 の出力はレジスタ125 に記憶される。セレクタ124 は図11に示すレジスタ81a が被除多項式として扱われている場合は逆元ＲＯＭ120 の出力を選択し、レジスタ81bが被除多項式として扱われている場合はレジスタ125 の出力を選択するように構成されている。また、シフトレジスタ126 より出力された0フラグデータはレジスタ128 で商多項式の係数データとタイミングが合わされて出力される。

【0221】いま、被除多項式の係数データがレジスタ81a に記憶されている場合について説明する。除算回路100 より出力された商データが0次のものは逆元ＲＯＭ120及びセレクタ122 へ入力され1次以上の商データはシフトレジスタ123 へ記憶される。なお、シフトレジスタ126 へは”Ｌ”が記憶されるものとする。また、逆元ＲＯＭ120 に入力されたデータが0の場合は本実施例では誤り検出として扱うように制御するものとする。（なお、上述の実施例でも同様に扱うものとする。）セレクタ124 は逆元ＲＯＭ120 の出力を、セレクタ122 は乗算回路121 の出力を、セレクタ127 はレジスタ125 の出力

を選択するように構成されている。この場合、商多項式の係数データに０が記憶されている場合でも、０／（逆元ＲＯＭ120 の出力）が実行され特に何も操作せずに正規化が実行される。これによりレジスタ81a に被除多項式の係数データが記憶されている場合の商多項式の正規化は実行される。

【０２２２】同様に、被除多項式の係数データがレジスタ81b に記憶されている場合について説明する。除算回路100 より出力された商データが０次のものは逆元ＲＯＭ120 及びセレクタ122 へ入力され、１次以上の商データはシフトレジスタ123 へ記憶される。その際、シフトレジスタ126 には次数検出回路88で検出された０フラグデータが記憶される。なお、本実施例ではレジスタ81bに記憶されている被除多項式の最大次数の係数データが０の場合”Ｈ”を出力する構成になっている。また、商多項式の０次の係数データが０の場合、本実施例では誤り検出として扱うように誤り訂正装置を制御するものとする。セレクタ124 はレジスタ125 の出力を選択するように制御される。セレクタ122 では、ユークリッド演算制御回路101 より出力される制御信号に基づき、まず初めは０次の商多項式の係数データがそのま出力され、それ以降の商多項式の係数を算出する際は乗算回路121 の出力を選択するように制御するように構成されている。セレクタ122 の出力はレジスタ125 に記憶される。これにより、初期値として商多項式の１次の係数データがセレクタ122 より出力された後、逐次乗算回路121 により $Q_i \times P_i$ が実行され商多項式がもとめられる。

【０２２３】一方、商多項式の係数データが０の場合（すなわち除算回路100 に除数として入力される被除多項式の係数データが０の場合）、除算回路100 では先ほども述べたように除算出力として１が出力され、それと同時に次数検出回路88より０検出フラグが出力される。よって、商多項式の係数データを求める際は、乗算回路121 では $Q_i \times 1$ が実行されレジスタ125 には Q_i が記憶されることになる。なお、シフトレジスタ126 より出力された０フラグデータはレジスタ128 で商多項式の係数データとタイミングが合わされてセレクタ127 に出力される。また、シフトレジスタ126 からは０フラグデータが出力され、セレクタ127 で０が選択される。そして、次の商多項式を算出する際は $Q_i \times P_{i+1}$ が実行される。

【０２２４】この構成によると、０フラグを記憶するシフトレジスタ126 とセレクタ122,124,127 とを上述のように制御することにより上述の演算を実行し商多項式の係数データを算出することが可能となる。よって、シフトレジスタ126 とセレクタ122,124,127 とを追加するのみの簡単な回路構成で誤り位置多項式を算出できる。

【０２２５】なお、第８実施例では、誤り位置多項式を算出する際の誤り位置多項式算出回路の商多項式を正規化する回路の構成を図20～22に示したが、逆元ＲＯＭ120 及び乗算回路121 部分を図20に示す除算回路100 と兼用し回路規模の縮小を図ってもよい。また、シフトレジスタ123 をレジスタ81a，81bと兼用して回路規模の縮小を図ってもよい。

【０２２６】また、第８実施例では、誤り位置多項式を算出する際の誤り位置多項式算出回路の回路構成を図20～22に示す構成をとってきたがこの構成に限るものではなく、図11または図14に示す回路構成をとる誤り訂正装置において、除算回路100 へ被除数として入力される多項式が、除多項式であった場合と被除多項式であった場合とで誤り位置多項式を算出する誤り位置算出回路での演算方法を切り換えるように構成すれば同様の効果を奏する。例えば、本実施例では商多項式の０次の係数データを１に正規化したが、商多項式の最大次数の係数を１に正規化する場合においても、商多項式の求め方の詳細は省略するが上述のように誤り位置算出手段中の商多項式を求める演算方法を切り換えれば同様に求められる。

【０２２７】また、第８実施例では、商多項式が２次の場合または除算回路100 より出力される除算結果が０の場合についても、図21または図22に示す回路構成により商多項式を算出したが、商多項式がある次数以上の場合または除算回路100 より出力された係数データに０の係数データが含まれていた場合、商多項式を算出せず誤り検出として処理し制御の簡略化及び回路規模の縮小を図ってもよい。

【０２２８】第９実施例．以下、本発明の第９実施例について説明する。本実施例では図11に示す誤り訂正装置を用いた場合について説明する。図23に第９実施例のブロック構成図を示す。本実施例では図11に示す誤り訂正装置のコア回路の構成にイレージャ訂正を行う際に必要となる修正シンドロームを算出するためのセレクタを追加した構成になっている。図において81a，81b，85～88，95，100 及び101 は図11と同一であるので説明は省略する。130,131,132 はセレクタ、133 はイレージャ位置を入力する入力端子である。なお、イレージャ訂正を行う際は、上記被除多項式及び除多項式を記憶するレジスタ81a，81bの数は、修正シンドロームの各係数データを記憶するため、（修正シンドロームの最大次数＋１）個のレジスタが被除数及び除数各々に必要となる。

【０２２９】イレージャ訂正とは誤りの位置はわかっているが誤り数値がわからないといった場合の誤り訂正に用いられる方法で、簡単に述べると、誤り位置情報より修正シンドロームを計算し、この修正シンドロームをもとに図25に示すユークリッドアルゴリズムを実行し誤り訂正を行う。その際、ユークリッドアルゴリズムにおいては終了条件を変えて、シンドローム多項式を修正シンドローム多項式に置き換えるだけで他は変わらない。（イレージャがある場合のユークリッドアルゴリズムの詳しい説明は前述の「符号理論」のpp.173～175 を参照）

【0230】上記修正シンドロームの算出方法を以下に簡単に説明する。いま、再生または受信データより算出されたシンドローム多項式S（Z）を

$S(Z)=S_{15}Z^{15}+S_{14}Z^{14}+S_{13}Z^{13}+\cdots S_1 Z+S_0$

とし、またイレージャの位置を$\alpha_0 \cdots \alpha_k$とする。この際、修正シンドロームS（z）'は

$S(z)'=S(z)\times\{(Z+\alpha_0)\times(Z+\alpha_1)\times\cdots\times(Z+\alpha_k)\}$

で算出される。

【0231】以下、図23を用いて修正シンドロームS（Z）'を求める際の回路動作を説明する。シンドローム生成が終了するとユークリッド演算のコア回路ではレジスタ81b にシンドローム多項式の各係数データがセットされる。その際、イレージャ数によりシンドローム多＊項式の最大次数の係数が記憶されるレジスタ位置が変わる。レジスタ81b にシンドローム多項式の係数データがセットされるとユークリッド演算制御回路101 より出力される制御信号によりセレクタ130 は誤り位置情報を、セレクタ131 はレジスタ81b の出力を、セレクタ132 はレジスタ81b に記憶されている１次次数が高いシンドローム多項式の係数データを選択するように制御される。そして、レジスタ81b の内容を右側にシフトすることにより

$S(Z)\times Z+S(Z)\times\alpha_0$

を実行する。（なお、α_0はイレージャの位置情報を示す。）数12に、具体的な筆算結果を示す。

【0232】

【数12】

$$\frac{\{S_{15}Z^{15}+S_{14}Z^{14}+S_{13}Z^{13}+\cdots+S_1 Z+S_0\}\times Z+\{S_{15}Z^{15}+S_{14}Z^{14}+S_{13}Z^{13}+\cdots+S_1 Z+S_0\}\times\alpha_0}{S'_{16}Z^{16}+S'_{15}Z^{15}+S'_{14}Z^{14}+\cdots S'_2 Z^2+S'_1 Z^1+S'_0}$$

【0233】なお、最大次数S_{15}については演算が行われても変わらないため係数をそのままシフトするように構成するものとする。また、０次の係数は$S_0\times\alpha_0$のみを実行するように構成する。そして、この動作を、逐次行うことにより修正シンドロームを算出する。図24に修正シンドロームを算出する際のセレクタ131，132の接続状態を示す。なお、この図は図23に示す回路の１部の接続状態を示したものである。図に示すように本実施例では修正シンドロームを算出する際はレジスタ81a は動作しない。

【0234】修正シンドロームの算出が終了すると図23に示す誤り訂正装置のコア回路はセレクタ130 を除算回路100 の出力を、セレクタ131 はレジスタ81a の出力を選択するように切り換えられる。また、セレクタ132 は同次数のレジスタ81b の出力を選択するように切り換えられ、各構成要素の接続が図11に示すものと同一の構成になる。一方、ユークリッド演算制御回路101 では、修正シンドロームを求める際、イレージャ数により図25に示す終了条件ｕを

$u=t$

から

$u=(2\times t+k)/2$

と設定する。なお、ｋはイレージャ訂正を行なうシンボルの数である。

【0235】ユークリッド演算が開始されるとまず始め、ユークリッド演算制御回路101 より出力されるユークリッド演算開始信号をもとにレジスタ81a に初期値である被除多項式Z^{2t}の係数が記憶されるとともに、レジスタ81b に除多項式である修正シンドローム多項式S（Z）'の各係数データが記憶される。また、次数検出回路87，88のカウンタにはそれぞれ次数の初期値である16と15とがセットされる。次数検出回路87，88内のカウンタは被除多項式の係数データを記憶しているレジスタ81が１回シフトされるたびにカウント値が１つ下がるように構成されている。それと同時に、除多項式である修正シンドローム多項式は次数検出回路88で最高次数の係数データが０であるか判定する。ユークリッド演算制御回路101 では、次数検出回路88で検出された除多項式（修正シンドローム多項式）の最大次数の係数が０の場合、除多項式の係数データが記憶されているレジスタ81ｂ の内容をシフトし（なお、この際加算回路86での加算動作を行わずレジスタ81b の内容をそのままシフトする構成になっている）最大次数のデータが０でなくなるまでシフト動作を繰り返す。その際、次数検出回路88中のカウンタはカウント値をシフト動作が行われるたびに１づつ減らされる。

【0236】また、第５実施例と同様に、被除多項式の係数データが記憶されているレジスタ81a はクロックが入力されると加算回路86の出力がラッチされるように構成されており、除多項式の係数データが記憶されているレジスタ81b は係数データが保持されるように構成されている。この動作を、被除多項式の次数が除多項式の次数より小さくなるまで繰り返す。

【0237】ユークリッド演算制御回路101 では、次数検出回路87（または88）より出力される上記剰余多項式の次数をもとに、剰余の次数が上記イレージャ訂正を行なう際に新たに設定されたユークリッド演算の終了条件を満たしていればユークリッド演算を終了し、終了条件

を満たしていなければ前回の演算で除多項式であった多項式を被除多項式として、また上記剰余多項式を除多項式として演算を実行する。従来例のように交換手段を有していないので回路の接続状態は変わらないが、レジスタ81a に供給されていたクロック１が止められて、反対にレジスタ81bにクロック２が供給される。よって、レジスタ81a の内容が保持され、レジスタ81b に剰余多項式の係数が逐次記憶される。

【０２３８】その際、ユークリッド演算回路中の除算回路では、次数検出回路88により被除多項式（剰余多項式）の最大次数が０であるかを判定し、０であった場合レジスタ81b に記憶されている被除多項式（剰余多項式）の内容をそのまま１つシフトするように制御信号を出力する。これは、除算回路100 へ出力される除数と被除数を格納するレジスタが決まっており被除多項式の係数データが除数として用いられる場合、被除多項式の最大次数の係数が０と成ってしまうと被除数（除多項式の最大係数）を０で割ることになり、解（出力）が不定となってしまう場合が生じるためである。なお、本実施例では除算回路100 の構成を図18に示す構成としているので最大次数の係数データが０の場合もあまり気にせず除算動作を行える。

【０２３９】上記例と同様に、被除多項式の剰余の次数が除多項式次数より小さくなるまで被除多項式の係数データを記憶しているレジスタの内容を演算しながらシフト動作を繰り返すことにより除算を実行する。そして、上記剰余多項式の次数を判定し所定の条件を満たしていなければレジスタ81a，81bに供給するクロックを切り換えてユークリッドアルゴリズムに基づく除算動作を再び行う。これにより、リード・ソロモン符号を復号する際に、イレージャ訂正をも考慮したユークリッドアルゴリズムに基づくガロア体上の多項式同士の除算を１多項式毎に行うことができ、誤り訂正処理速度を向上させることが出来るとともに交換手段（セレクタ）を持たないため回路規模の縮小が図れ、またセレクタ制御も必要なくなる。

【０２４０】上記実施例５～９では光ディスク等に採用されている最小ハミング距離17のリード・ソロモン符号について述べたがこれに限るものではなく最小ハミング距離が17以外の符号でも同様の効果を奏する。また、上記実施例に示す回路構成にて、最小ハミング距離が異なる２種類以上のリード・ソロモン符号も復号できる。例えば、積符号形式のリード・ソロモン符号の復号も同一の回路でユークリッドアルゴリズムの初期値及び終了条件をかえるだけで復号することができる。

【０２４１】また、各実施例では誤り訂正符号としてリード・ソロモン符号を用いた場合について説明したがＢＣＨ符号などの他の線形誤り訂正符号の復号に上記除算回路を用いて復号しても同様の効果を奏する。

【０２４２】また、各実施例では除算演算時、レジスタ81a，81bのシフト、及びデータの保持をクロック１，２を制御することにより実行したがこれに限るものではなく、例えばレジスタ81a，81bの入力にセレクタを設け、セレクタにより除多項式側のレジスタは自分自身の出力をラッチし、被除多項式側のレジスタは加算回路86の出力をラッチする様に制御してもよい。

【０２４３】また、被除多項式と除多項式との交換判定は次数検出回路87，88より出力される各多項式の次数差を検出する方法に限るものではない。例えば、被除多項式の次数と除多項式の次数との差を除算を始める前に予め求めておき、この情報をもとに一義的に除算回路のシフト回数を決定し次数差を検出せず除算を行うなどの方法を用いてもよい。

【０２４４】また、各実施例では、シンドローム多項式，剰余多項式等の次数を検出する際、レジスタ81a，81bにおける最大次数の係数が記憶されているレジスタのみに次数検出回路87，88を接続し検出していたが、これに限るものではなく、例えば、各係数データを記憶するレジスタ81に係数判定回路（係数データが０であるか否かを判定する回路）を設けて次数判定を行ってもよい。

【０２４５】更に、詳細は省略するがユークリッドアルゴリズム以外、例えば剰余復号等にも上記回路構成を適用できることは言うまでもない。

【０２４６】

【発明の効果】以上のように第１発明の誤り訂正装置は、入力信号にあらかじめ付加されている線形符号である誤り訂正符号によって符号化された信号中に発生する誤りを訂正するための誤り訂正装置にあって、誤り訂正符号が付加された単位情報ブロックより再生または受信信号中に含まれる誤りにのみ依存するシンドロームを生成するためのシンドローム生成手段と、生成されたシンドロームを基に誤り位置多項式σ（ｚ）及び誤り数値多項式ω（ｚ）を算出する誤り位置・数値多項式演算手段と、算出された誤り位置多項式σ（ｚ）を基に誤り位置を算出する誤り位置算出手段と、上記算出された誤り数値多項式ω（ｚ）及び誤り位置情報を基に誤り訂正を行う誤り訂正手段とを有し、誤り訂正時に、少なくともシンドローム生成手段、誤り位置・数値多項式演算手段及び誤り位置算出手段の３つの処理動作を並行して動作するように構成されているので、例えば図33に示す１誤り訂正ブロックに復号処理を施す際２倍以上のスピードで処理を完了することが可能となる。

【０２４７】また第２発明の誤り訂正装置は、比較的高速化が図れるシンドローム生成手段及び誤り位置算出手段を駆動するクロックを、誤り位置・数値多項式演算手段を駆動するクロックより高い周波数のクロックで駆動するように構成したので、１誤り訂正ブロックを復号するのに要する復号時間を短縮することができ、処理速度を上げることが可能となる。

【０２４８】また第３発明の誤り訂正装置は、比較的計

算にステップ数を要するシンドローム生成手段及び誤り位置算出手段のステップ数にあわせて誤り位置・数値多項式演算手段のステップ数を増やすことにより回路規模を図ったので、誤り訂正装置全体の復号速度を損ねることなく回路規模の削減を図れる。

【０２４９】また第４発明の誤り訂正装置は、誤り訂正手段によりＲＡＭ内のデータに誤り訂正を施す際に誤り位置算出手段より出力される誤り位置情報に基づく誤ったデータを逐次読み出した後に、誤り訂正を施したデータを逐次ＲＡＭに書き込むように構成したので、通常１シンボルの誤り訂正を行う際、ＲＡＭからのデータの読み出し、誤り訂正、ＲＡＭへのデータの書き込みと最低３ステップかかっていた動作を２ステップで行うことが可能となり、誤り訂正装置の高速化が図れる。よって、高速化のネックとなるＲＡＭのデータのアクセスを常に行うように構成されているので、回路規模の増大を極力控えて最高速の誤り訂正装置を構成することが可能となる。

【０２５０】また第５発明の誤り訂正装置は、所定の被除多項式を除多項式で除算し、さらにその剰余を除多項式として、また前記除多項式を被除多項式として除算して剰余の多項式の次数がある条件を満たすまで上記動作を繰り返すユークリッド互除の処理過程を含んでリード・ソロモン符号による誤り訂正処理を行う誤り訂正装置において、上記被除多項式及び除多項式の各係数データを記憶しておくレジスタをシフトレジスタ状に配置し、そして、上記２つの多項式の最大次数の係数同士を除算する除算回路と、除算回路の除数側に設定された多項式の係数データと除算回路の出力を乗算する乗算回路と乗算回路の出力と除算回路の被除数側に設定された多項式の係数データを加算する加算回路とを配置し、ガロア体上の除算を実行する際に被除多項式の係数データが記憶されているレジスタの内容を逐次シフトすることによりユークリッド互除過程に基づく除算を実行するように構成しているので、ユークリッド互除過程におけるガロア体上の多項式同士の除算を実行する際、各多項式の係数データを交換するためのセレクタをもたずにユークリッド互除過程中の除算を実行するため、回路規模の縮小を図れるとともにセレクタ制御が不必要になり回路の簡素化が図れる。

【０２５１】また第６発明の誤り訂正装置は、乗算回路をユークリッド互除過程における初期値Z^{2t}をセットするレジスタ群に配置し乗算回路の数を削減するように構成したので、除算動作を行うため乗算回路の数を１つ減らすことができ、前述したように約 400ゲート程度の回路規模の縮小が図れる。

【０２５２】また第７発明の誤り訂正装置は、除算回路において、除数として入力される多項式の最大次数の係数が０であった場合、除算回路の出力を１として出力するように構成したので、ユークリッド演算制御回路で被除多項式または除多項式の最大次数が０であった場合でも、特別な制御を加えることなくユークリッドアルゴリズムにおけるガロア体上の多項式同士の演算を実行でき、制御回路の簡素化が図れる。

【０２５３】また第８発明の誤り訂正装置は、除算回路の出力を用いて誤り位置多項式σ（Ｚ）を算出する際、除算回路に入力される被除多項式の最大次数の係数データが除数として扱われる場合と被除数として扱われる場合とで算出方法を切り換えるように構成されているので、誤り位置多項式が正確に算出されるとともに、復号ステップ数を削減できる。

【０２５４】また第９発明の誤り訂正装置は、除算手段の出力と誤り位置情報を選択する第１の選択手段と、除算手段の除数側の多項式の係数を記憶しておく記憶手段の出力と、除算手段の被除数側の多項式の係数データを記憶しておく記憶手段の出力とを選択する第２の選択手段と、除算手段の除数側の多項式の係数とそれより１次次数が高い除算手段の除数側の多項式の係数とを選択する第３の選択手段と、第１の選択手段の出力と第３の選択手段の出力とを乗算する乗算手段と、乗算手段の出力と第２の選択手段の出力とを加算する加算手段とを有し、消失訂正時に修正シンドローム多項式を演算する際、第１，第２及び第３の選択手段でそれぞれ誤り位置情報，上記除算手段の除数側の多項式の係数データ及び１次次数が高い多項式の係数を選択するように構成しているので、イレージャ訂正を行う際に算出する修正シンドローム多項式の算出をセレクタ回路を付加するだけで行うことができ、回路規模の増大を極力抑えてイレージャ訂正にも対応できる。

【図面の簡単な説明】

【図１】本発明の誤り訂正装置のブロック構成図である。

【図２】図１に示す誤り訂正装置の回路動作を説明するためのタイミングチャートである。

【図３】本発明の他の誤り訂正装置のブロック構成図である。

【図４】図３に示す誤り訂正装置の回路動作を説明するためのタイミングチャートである。

【図５】ユークリッド演算回路のガロア体上の多項式の除算回路のブロック構成図である。

【図６】図５に示す除算回路の回路動作を説明するためのタイミングチャートである。

【図７】誤り訂正装置全体の動作を説明するためのタイミングチャートである。

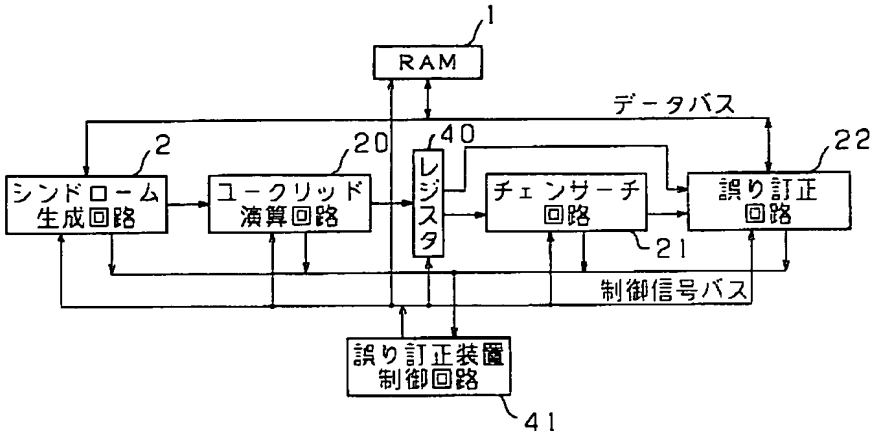
【図８】ユークリッド演算回路のガロア体上の多項式の他の除算回路のブロック構成図である。

【図９】誤り訂正回路の回路動作を説明するためのタイミングチャートである。

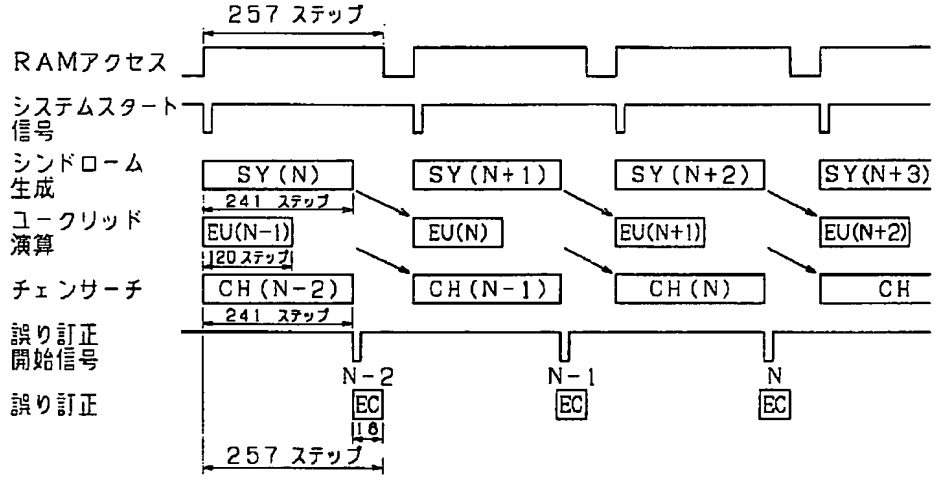
【図１０】誤り訂正回路の回路動作を説明するための他のタイミングチャートである。

【図１１】本発明の誤り訂正装置のユークリッド演算を行う際のガロア体上の多項式同士の除算を行う他の除算回路のブロック構成図である。
【図１２】図１１に示す除算回路の回路動作を説明するための図である。
【図１３】図１１に示す除算回路の回路動作を説明するための図である。
【図１４】本発明の誤り訂正装置のユークリッド演算を行う際のガロア体上の多項式同士の除算を行う他の除算回路のブロック構成図である。
【図１５】図１４に示す除算回路の回路動作を説明するための図である。
【図１６】図１４に示す除算回路の回路動作を説明するための図である。
【図１７】図１４に示す除算回路の回路動作を説明するための図である。
【図１８】本発明の誤り訂正装置における除算回路のブロック構成図である。
【図１９】本発明の誤り訂正装置における加算回路のブロック構成図である。
【図２０】本発明の誤り訂正装置における誤り位置算出手段の回路構成図である。
【図２１】本発明の誤り訂正装置における他の誤り位置算出手段の回路構成図である。
【図２２】本発明の誤り訂正装置における更に他の誤り位置算出手段の回路構成図である。
【図２３】イレージャ訂正を考慮した本発明の誤り訂正装置のユークリッド演算を行う際のガロア体上の多項式同士の除算を行う除算回路のブロック構成図である。
【図２４】本発明の誤り訂正装置の回路動作を説明するための回路接続図である。
【図２５】ユークリッドアルゴリズムの動作を説明するフローチャートである。
【図２６】従来の誤り訂正装置のブロック構成図である。
【図２７】シンドローム生成回路のブロック構成図である。
【図２８】従来の誤り訂正演算回路のブロック構成図である。
【図２９】従来のユークリッド演算回路のガロア体上の多項式の除算回路のブロック構成図である。
【図３０】チェンサーチ回路のブロック構成図である。
【図３１】誤り訂正符号による符号化及び復号化の手順を示すフローチャートである。
【図３２】家庭用ディジタルＶＴＲに採用されている誤り訂正符号の構成図である。
【図３３】図３２に示す誤り訂正符号の記録方向の１ラインの構成図である。
【図３４】従来の誤り訂正装置の動作を説明するためのタイミングチャートである。
【図３５】従来の誤り訂正装置の回路動作を説明するためのタイミングチャートである。
【図３６】Fig.12は従来の誤り訂正装置の動作を説明するためのタイミングチャートである。
【図３７】従来の誤り訂正装置のユークリッド演算を行う際のガロア体上の多項式同士の除算を行う除算回路のブロック構成図である。
【図３８】従来のガロア体上の多項式同士の除算を行なうための除算回路動作を説明するための図である。
【図３９】従来のガロア体上の多項式同士の除算を行なうための除算回路動作を説明するための図である。
【図４０】従来の誤り訂正装置のユークリッド演算を行う際のガロア体上の多項式同士の除算を行う除算回路動作を説明するための図である。
【図４１】従来の誤り訂正装置のユークリッド演算を行う際のガロア体上の多項式同士の除算を行う除算回路動作を説明するための図である。
【符号の説明】
1　ＲＡＭ
2　シンドローム生成回路
20　ユークリッド演算回路
21　チェンサーチ回路
22　誤り訂正回路
41　誤り訂正装置制御回路
81a，81b　レジスタ
85　乗算回路
86　加算回路
100　除算回路
101　ユークリッド演算制御回路
130　セレクタ
131　セレクタ
132　セレクタ

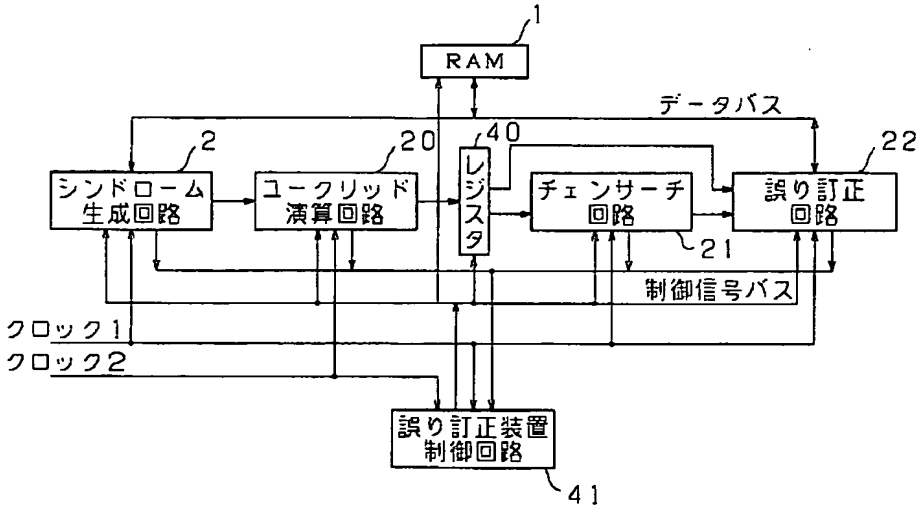
【図1】



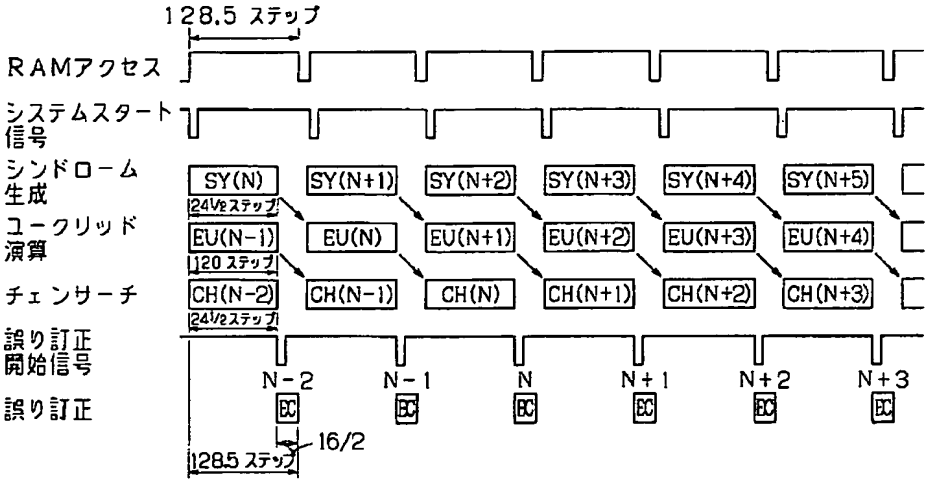
【図2】



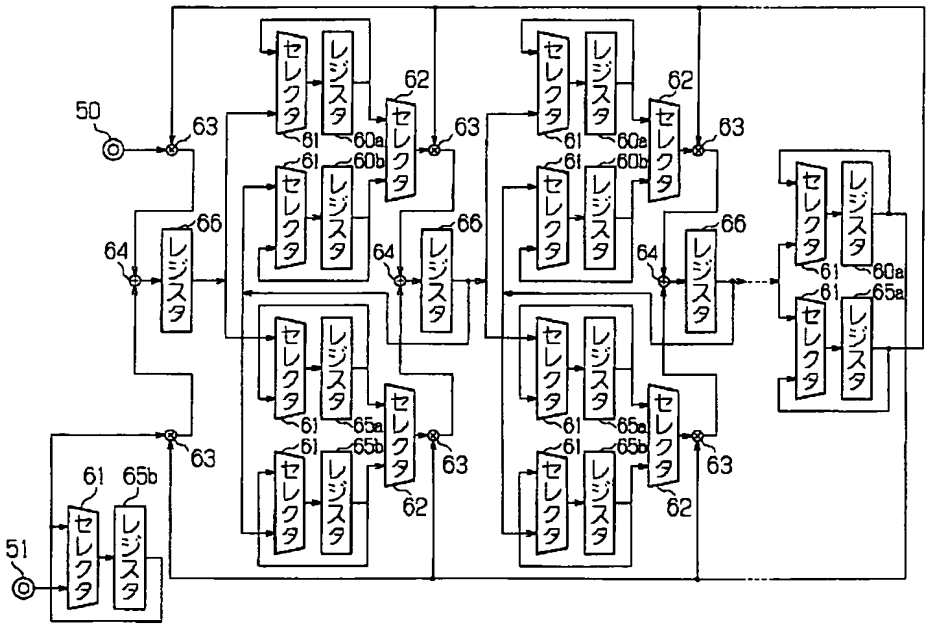
【図3】



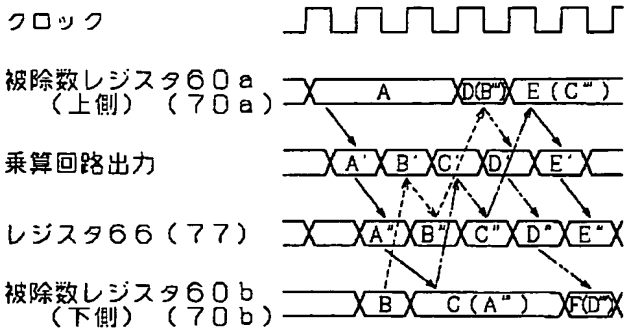
【図４】


128.5 ステップ
RAMアクセス
システムスタート信号
シンドローム生成
ユークリッド演算
チェンサーチ
誤り訂正開始信号
誤り訂正
SY(N)
SY(N+1)
SY(N+2)
SY(N+3)
SY(N+4)
SY(N+5)
24½ステップ
EU(N-1)
EU(N)
EU(N+1)
EU(N+2)
EU(N+3)
EU(N+4)
120 ステップ
CH(N-2)
CH(N-1)
CH(N)
CH(N+1)
CH(N+2)
CH(N+3)
24½ステップ
N-2
N-1
N
N+1
N+2
N+3
EC
16/2
128.5 ステップ

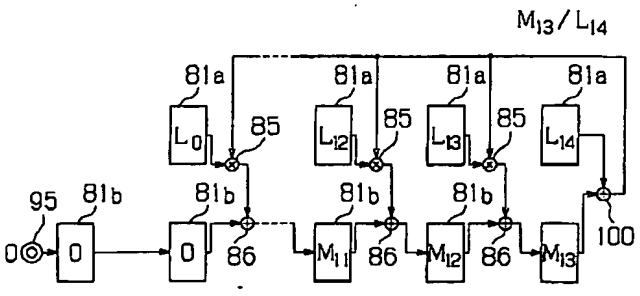
【図５】


50
51
63
64
66
62
61
60a
60b
65a
65b
セレクタ
レジスタ

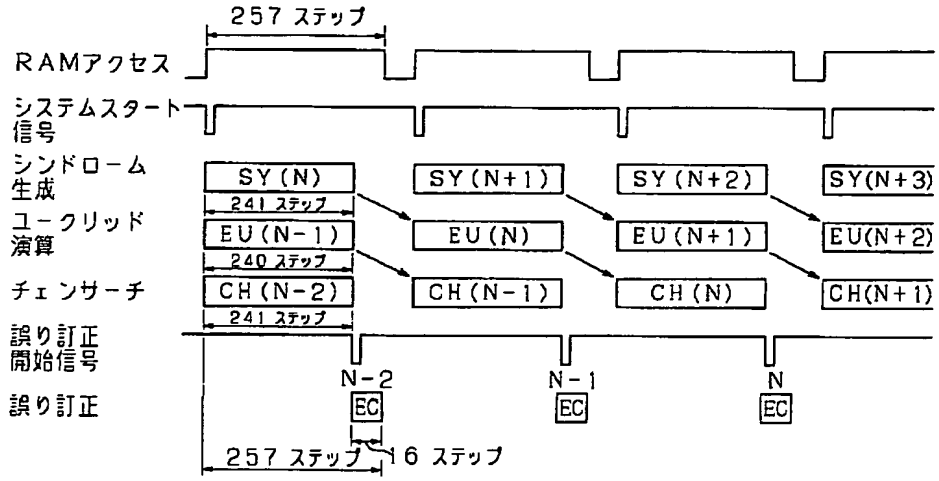
【図６】


クロック
被除数レジスタ６０ａ（上側）（７０ａ）
乗算回路出力
レジスタ６６（７７）
被除数レジスタ６０ｂ（下側）（７０ｂ）
A
D(B‴)
E(C‴)
A'
B'
C'
D'
E'
A"
B"
C"
D"
E"
B
C(A‴)
F(D‴)

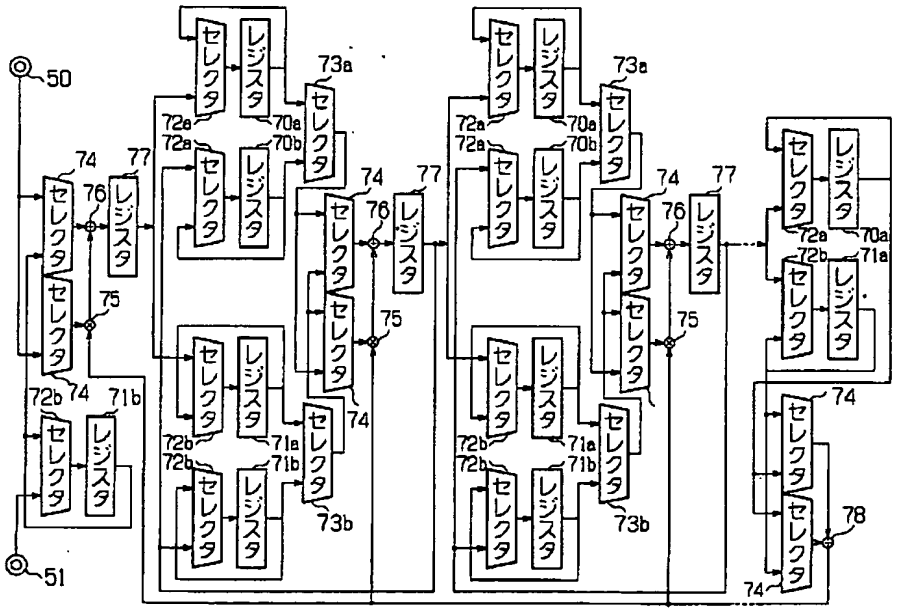
【図１７】


M13/L14
81a
81b
85
86
95
100
0
L0
L12
L13
L14
M11
M12
M13

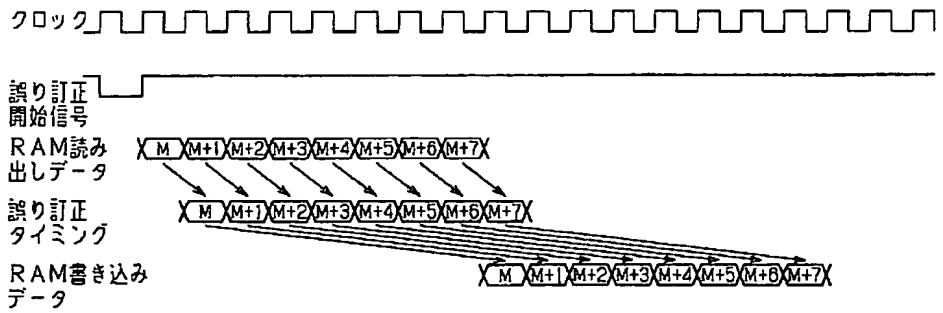
【図7】



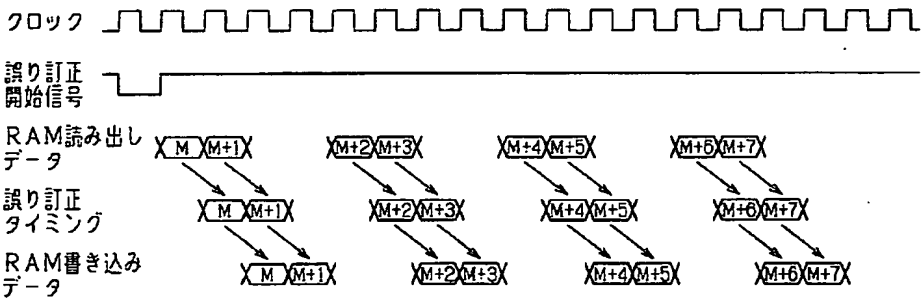
【図8】



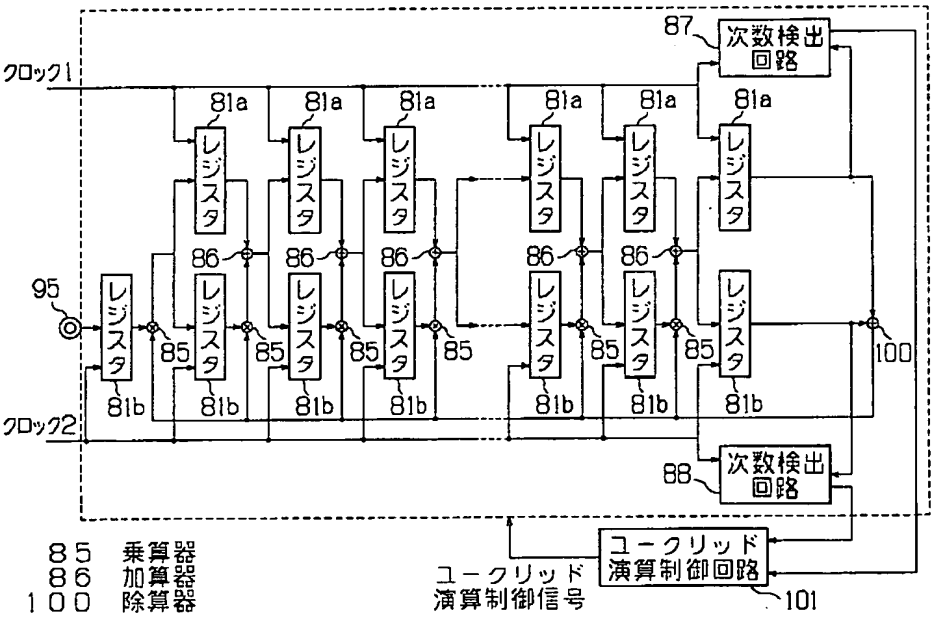
【図9】



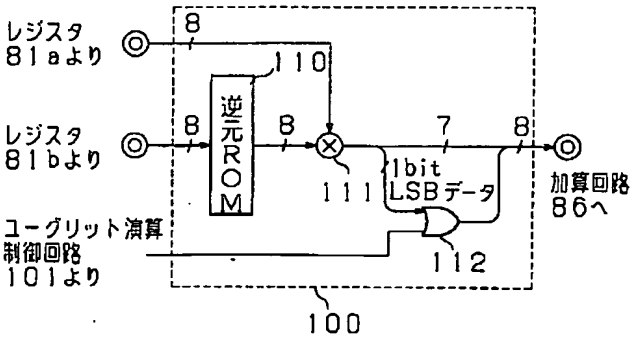
【図１０】



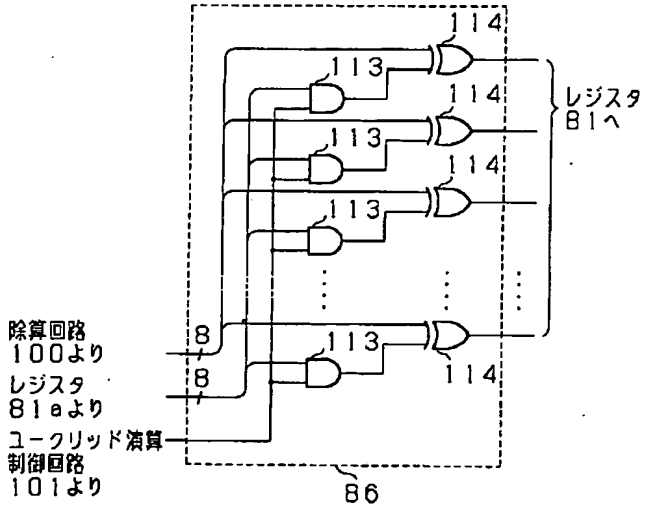
【図１１】



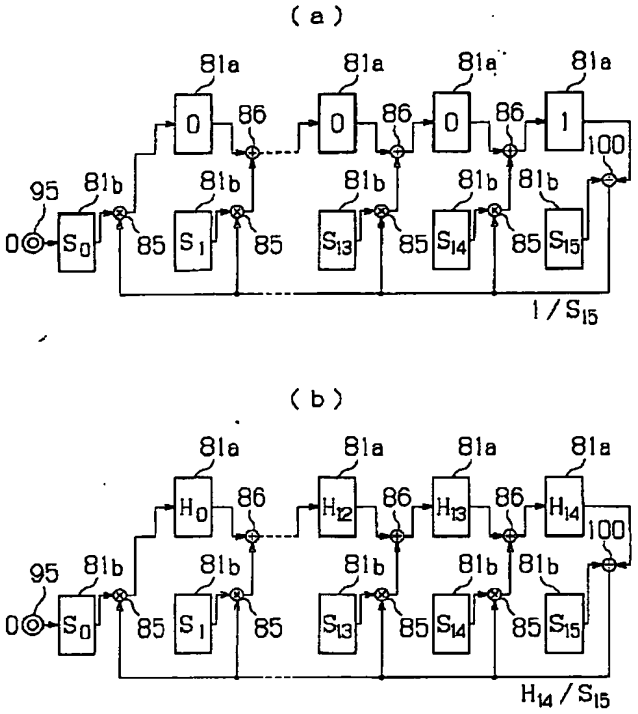
【図１８】



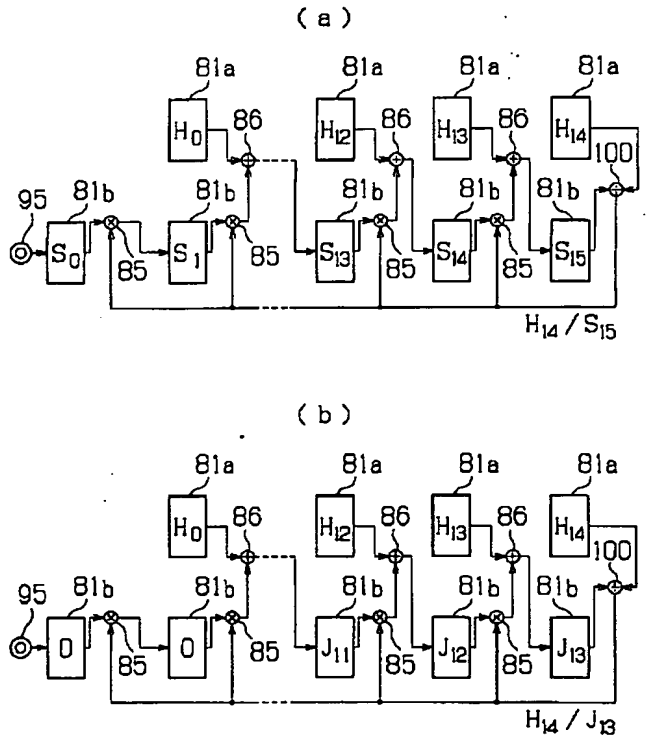
【図１９】



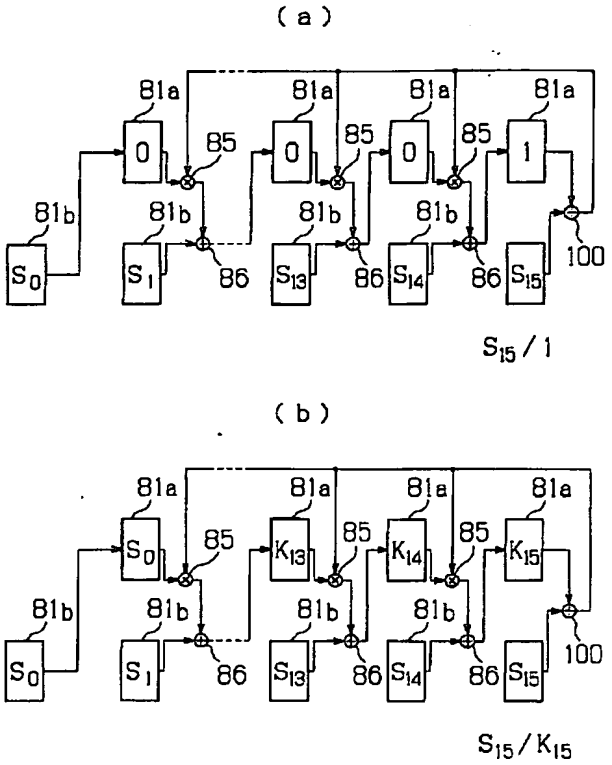
【図１２】



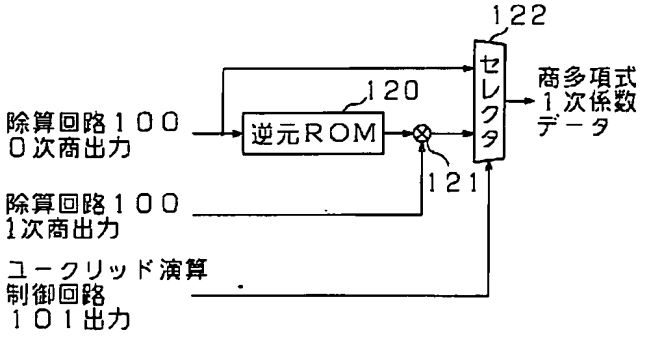
【図１３】



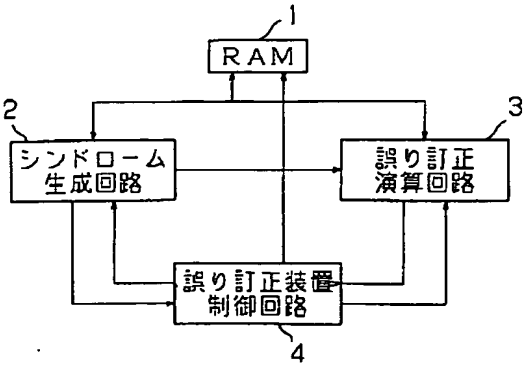
【図１５】



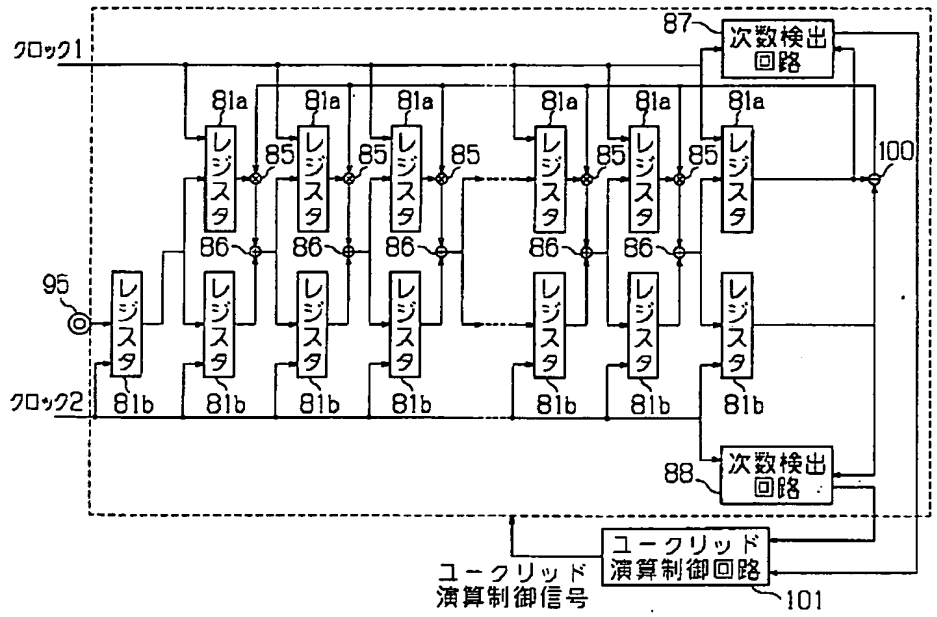
【図２０】



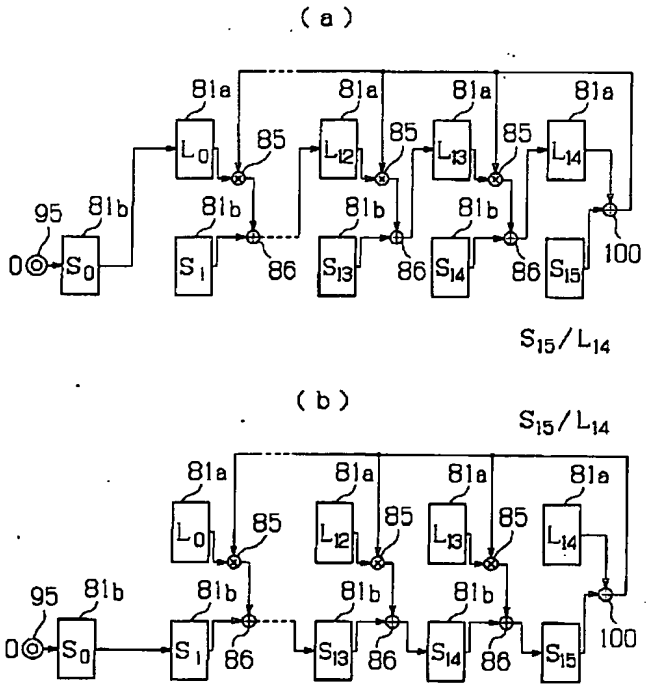
【図２６】



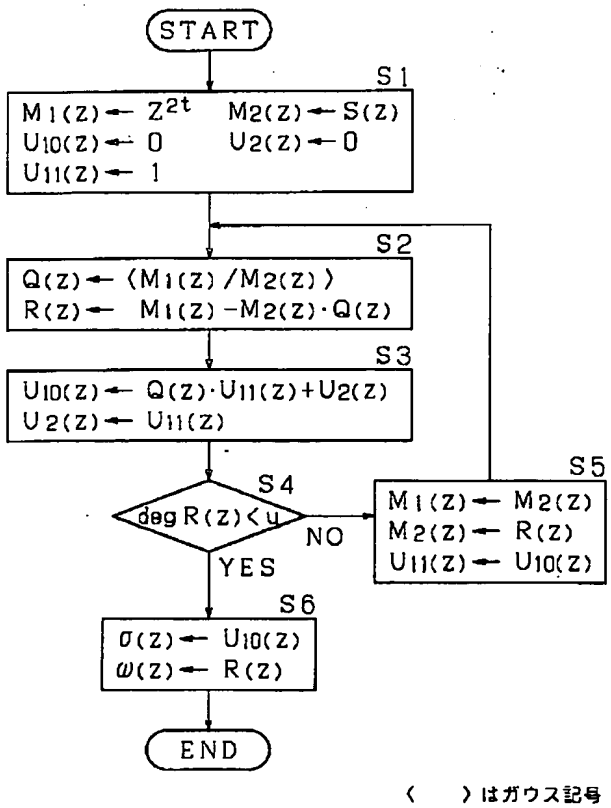
【図１４】



【図１６】

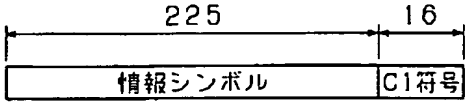


【図２５】

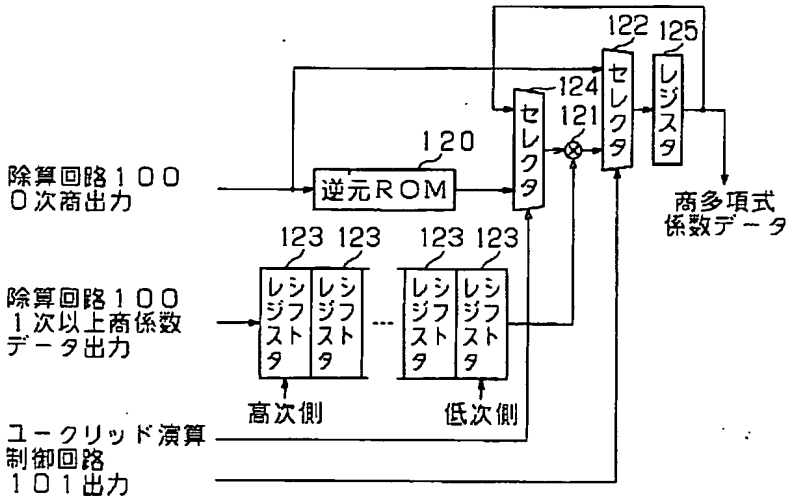


( ) はガウス記号

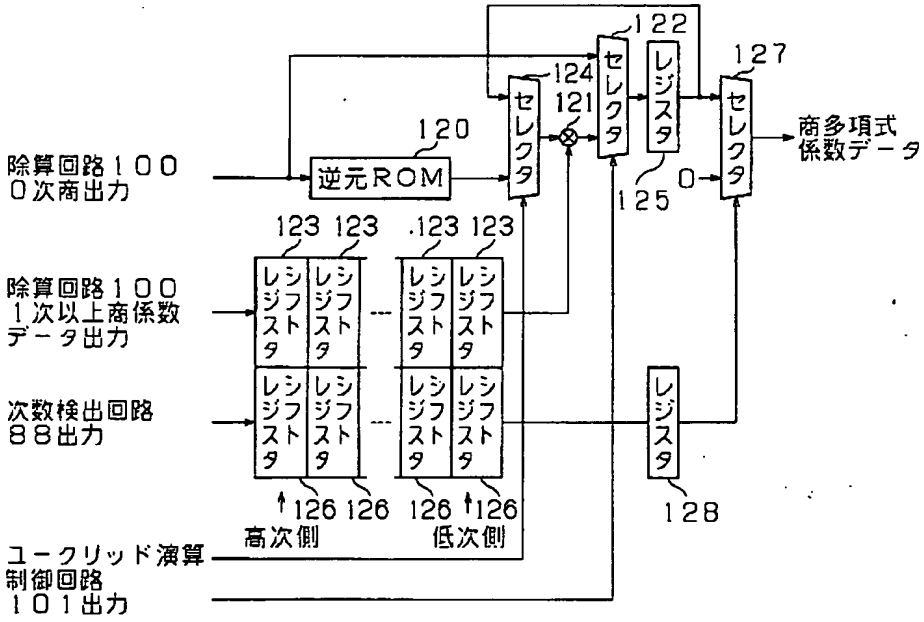
【図３３】



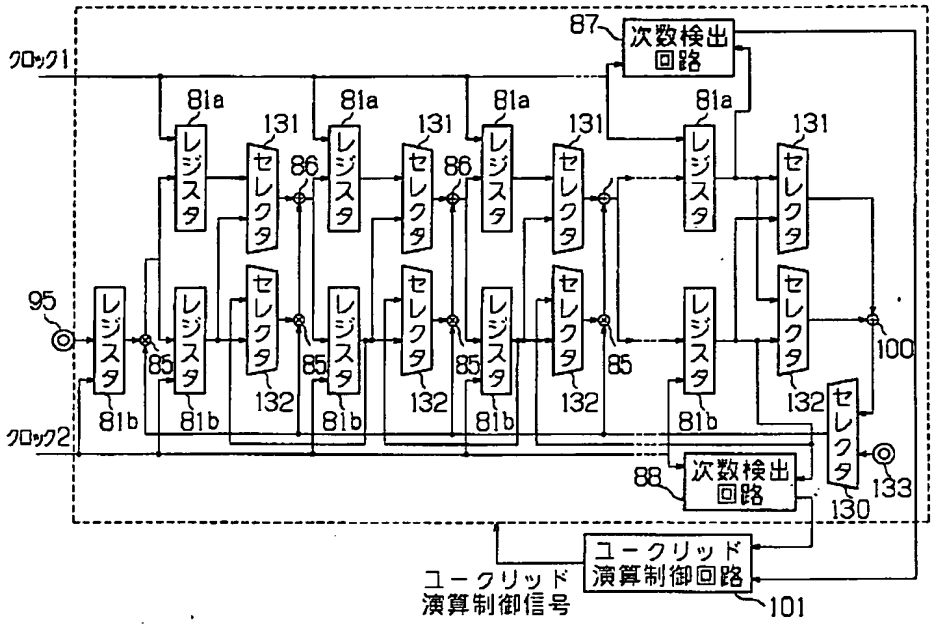
【図２１】



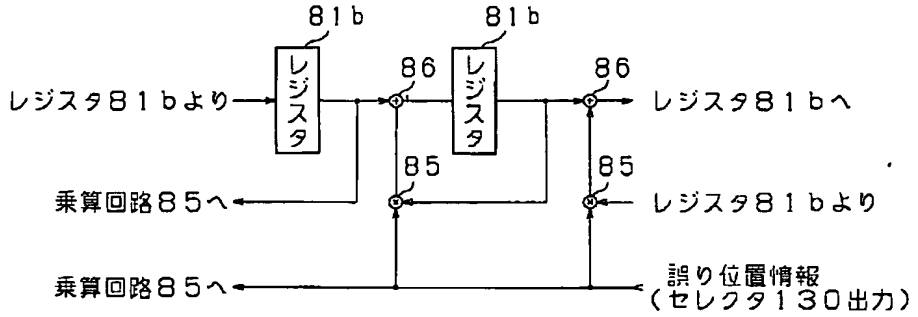
【図２２】



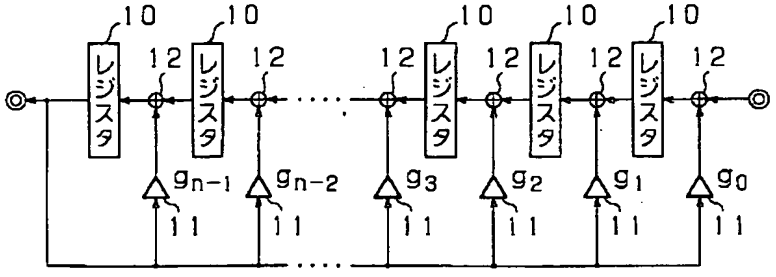
【図２３】



【図２４】

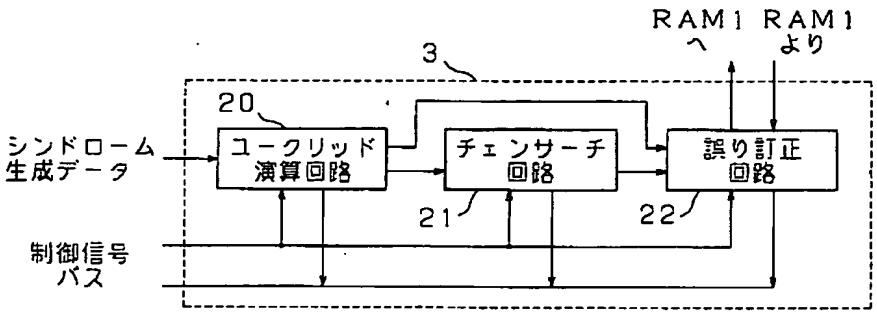


【図２７】

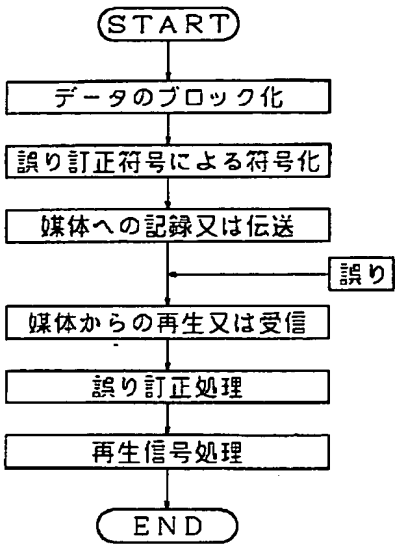


g_i（$i=1,2,\cdots,n-1$）：は乗算定数

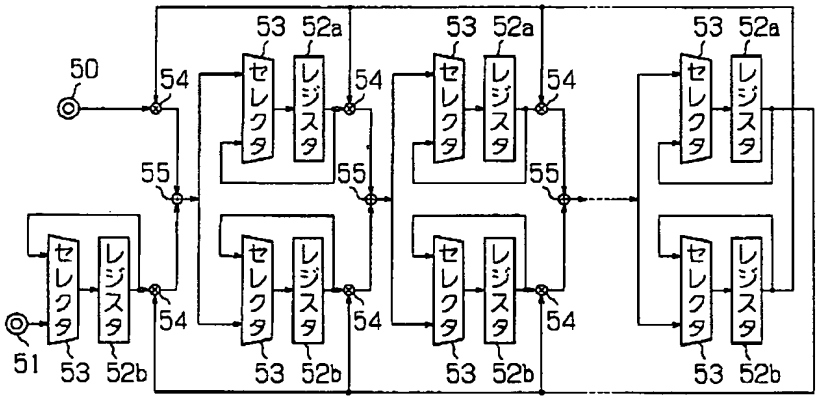
【図２８】



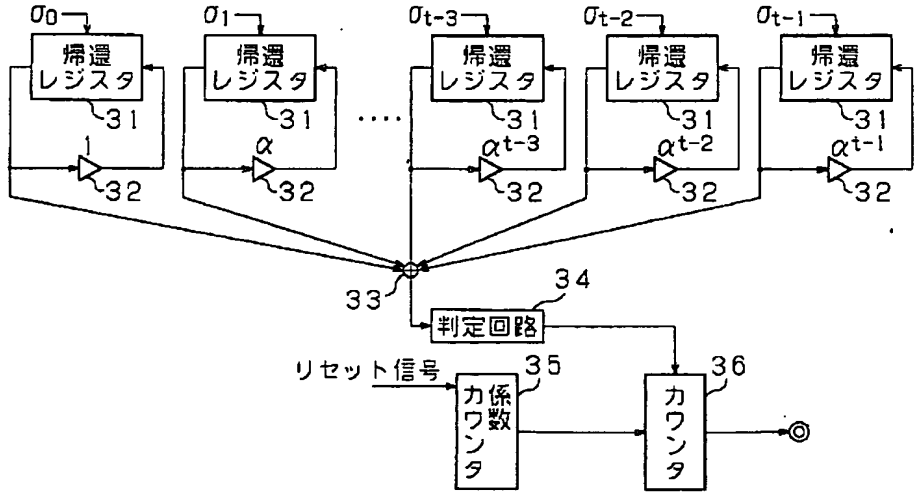
【図３１】



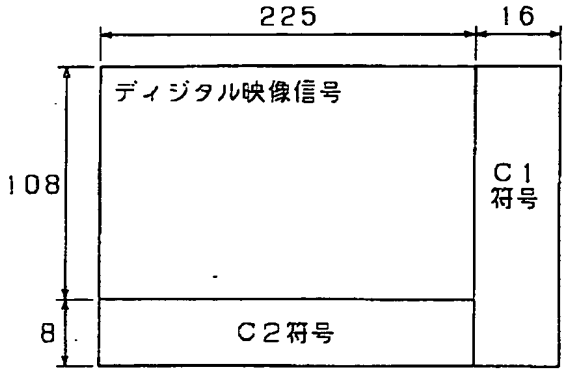
【図２９】



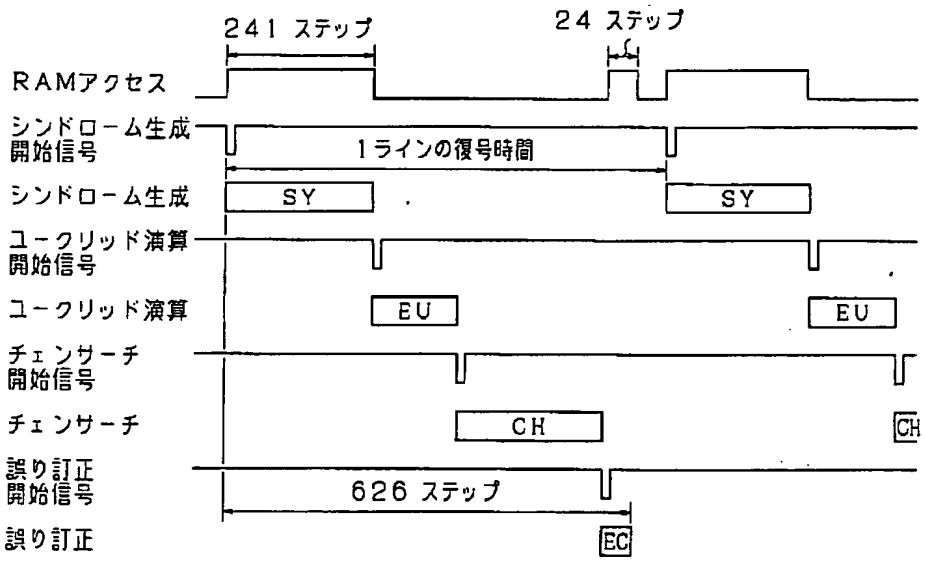
【図３０】



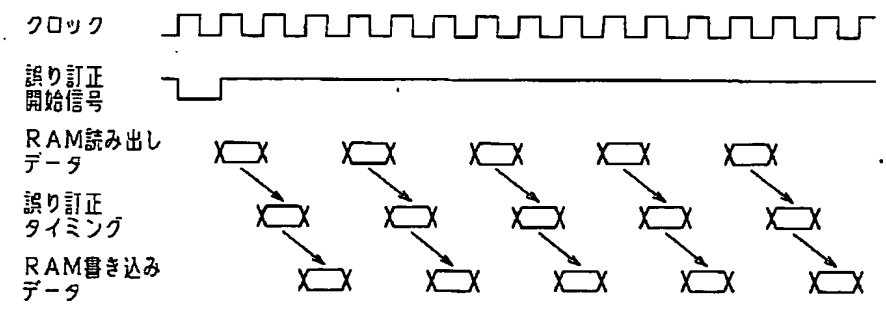
【図３２】



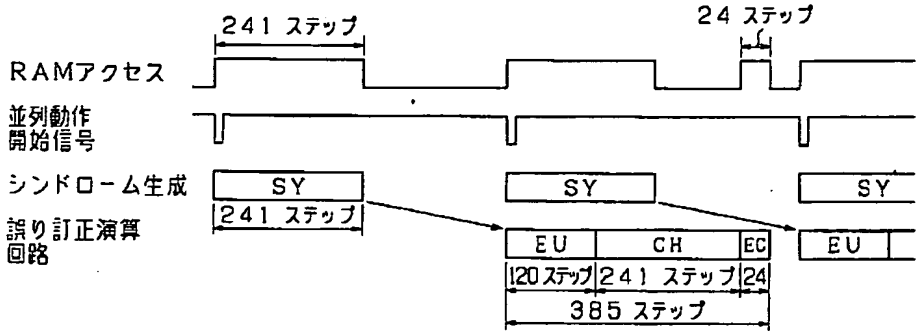
【図３４】



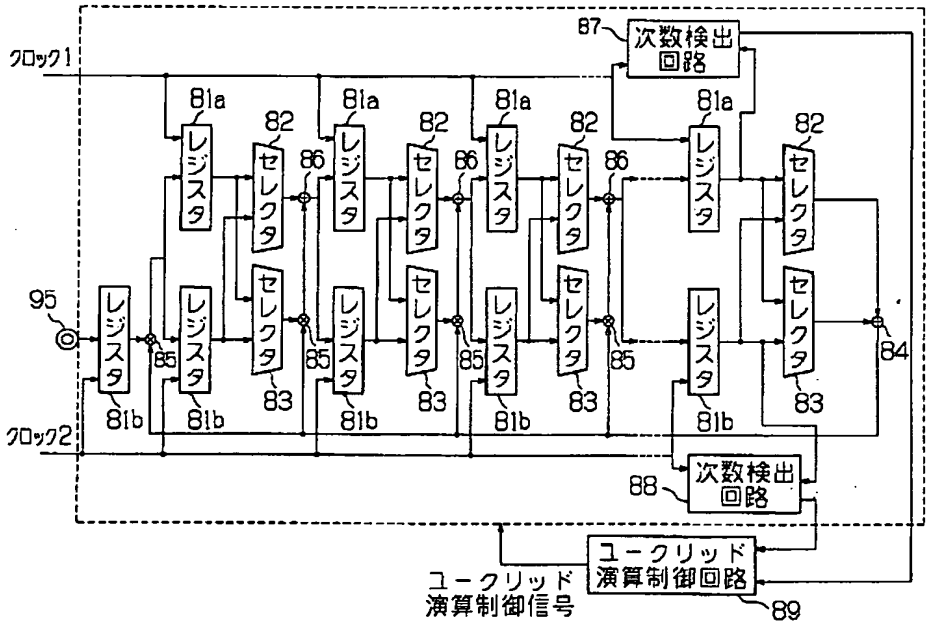
【図３５】



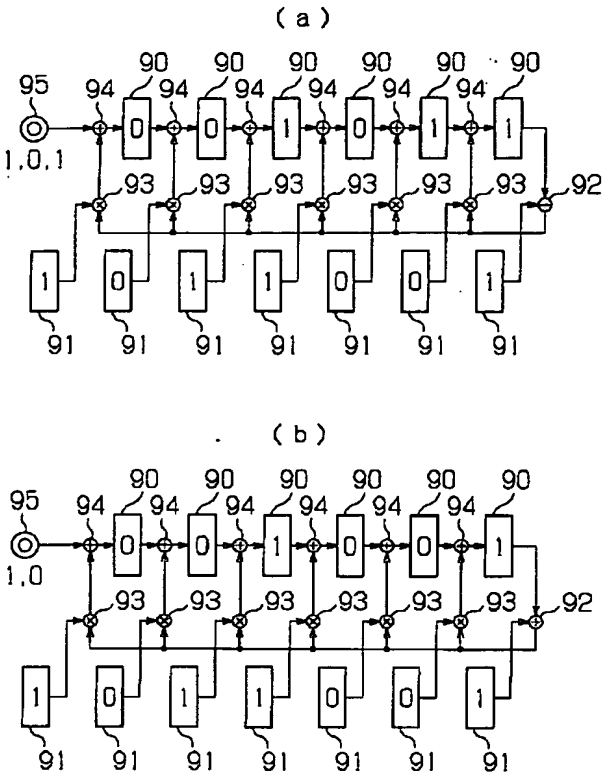
【図３６】


241 ステップ
24 ステップ
RAMアクセス
並列動作
開始信号
シンドローム生成
SY
241 ステップ
誤り訂正演算
回路
EU
CH
EC
120ステップ
241 ステップ
24
385 ステップ

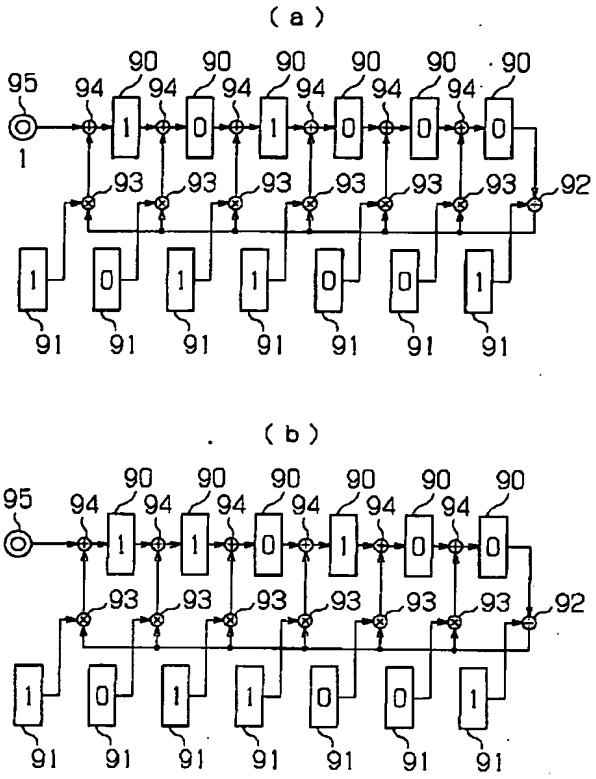
【図３７】


クロック1
クロック2
95
81a
81b
82
83
85
86
84
87
次数検出回路
88
次数検出回路
ユークリッド演算制御回路
89
ユークリッド演算制御信号
レジスタ
セレクタ

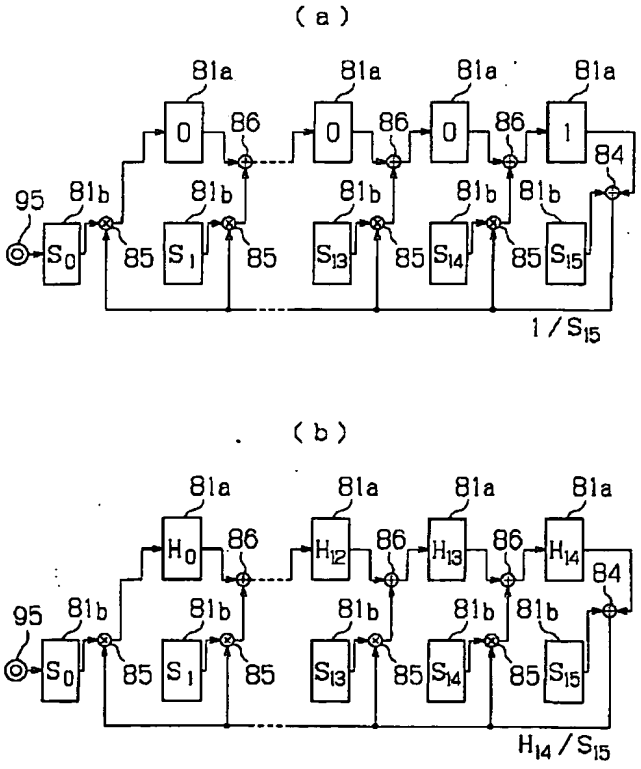
【図３８】



【図３９】

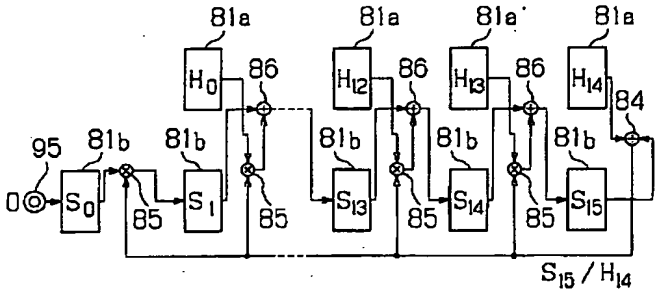


【図４０】

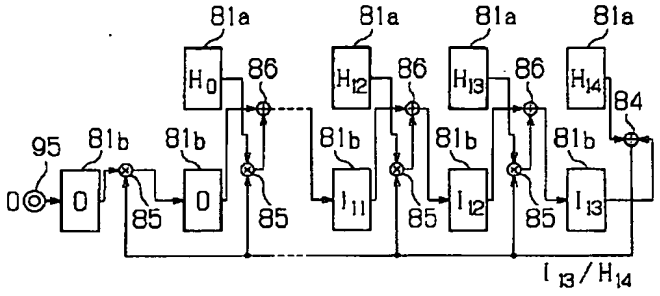


【図４１】

（ａ）



（ｂ）



フロントページの続き

(72)発明者　石本　順子
京都府長岡京市馬場図所１番地　三菱電機
株式会社電子商品開発研究所内